# ENVELOPE: ESTIMATION OF BOTTLENECK AND AVAILABLE BANDWIDTH OVER MULTIPLE CONGESTED LINKS

A Thesis

by

AMIT BHATI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2004

Major Subject: Computer Science

ENVELOPE: ESTIMATION OF BOTTLENECK AND AVAILABLE BANDWIDTH OVER MULTIPLE CONGESTED LINKS

A Thesis

by

AMIT BHATI

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

| | |
|---|---|
| Dmitri Loguinov<br>(Chair of Committee) | Donald K. Friesen<br>(Member) |
| A. L. Narasimha Reddy<br>(Member) | Valerie E. Taylor<br>(Head of Department) |

December 2004

Major Subject: Computer Science

# ABSTRACT

Envelope: Estimation of Bottleneck and Available Bandwidth over Multiple Congested Links. (December 2004)

Amit Bhati, B.Tech., Kakatiya University, India

Chair of Advisory Committee: Dr. Dmitri Loguinov

Bandwidth estimation has been extensively researched in the past. The majority of existing methods assume either negligible or fluid cross-traffic in the network during the analysis. However, on the present-day Internet, these assumptions do not always hold right. Hence, over such paths the existing bandwidth estimation techniques become inaccurate. In this thesis, we explore the problem assuming arbitrary cross-traffic and develop a new probing method called Envelope, which can simultaneously estimate bottleneck and available bandwidth over an end-to-end path with multiple heavily congested links. Envelope is based on a recursive extension of the stochastic queuing model first proposed by Kang, Liu, Dai and Loguinov (2004), and a modified packet-train methodology. We use two small packets to surround the probing packet-trains and preserve the inter-packet spacing of probe traffic at each router in the path-suffix. The preserved spacings are then used by the receiver to estimate bandwidth. We first reproduce results for a single congested router case using the model proposed by Kang *et al.* Next, we extend it to the case of multiple congested routers with arbitrary cross-traffic and develop the methodology Envelope. We evaluate the performance of Envelope in various network path topologies and cross-traffic conditions through extensive NS-2 simulations. We also evaluate various probe-traffic parameters which affect the accuracy of this method and obtain the range of values for these parameters that provide good estimation results. Finally, we compare the bandwidth estimation results of our method with the results of other existing meth-

ods such as IGI (2003) , Spruce (2003), Pathload (2002), and CapProbe (June 2004) using simulation in Network Simulator (NS-2) with varied network topologies and cross-traffic.

To my parents for always supporting me in tough times

# ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, Dr. Loguinov, for always being a source of inspiration and motivation and for giving a definite direction to my work. I would also like to thank my committee members, Dr. Friesen and Dr. Reddy, for invaluable suggestions for my thesis. Last but not least, I would like to thank all those who patiently edited my thesis.

TABLE OF CONTENTS

CHAPTER Page

# LIST OF FIGURES

FIGURE Page

# LIST OF TABLES

# CHAPTER I

# INTRODUCTION

Use of the Internet in various network applications such as audio-video streaming, web/distributed database application, mobile computing and multicasting is widespread. Due to this, it is imperative that the data flow in Internet is fast with less delays and high QoS characterized by low data loss and high throughput. In recent years, the onus of a quality service has shifted to underlying network application because of significant improvement in Internet servers [1], [2], [3], [4]. One of the factors which can considerably improve the QoS of these applications is the knowledge of available or bottleneck bandwidths of end-to-end network paths. Therefore, a lot of effort has been put in the past as well as current Internet studies to develop new techniques which can provide better estimates of these bandwidths. In this thesis, we develop Envelope, a new end-to-end bandwidth estimation method that can simultaneously measure bottleneck capacity and available bandwidth of network paths with multiple congested links.

## A. Bandwidth Estimation Metrics

In the perspective of data/packet networks, bandwidth is defined as the amount of data that can be transferred over the network in a fixed amount of time. It is usually expressed in terms of bits per second (bps) or in higher units like Mbps (millions of bits per second). A study by Prasad *et al.* in [5], classifies bandwidth of a network path according to various throughput-related concepts. The authors identify three different metrics of bandwidth in the context of links at the IP layer: capacity,

---

The journal model is *IEEE Transactions on Communications.*

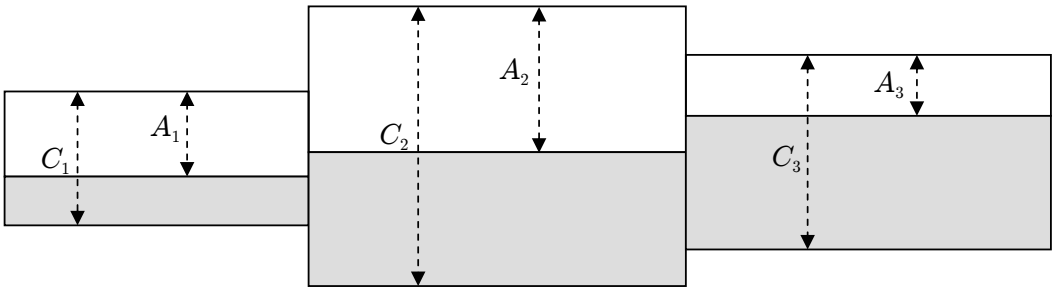



Fig. 1. Three-hop pipe model with fluid cross-traffic.

available bandwidth and bulk transfer capacity (BTC). In this thesis, we focus on the estimation of the first two metrics for both individual links and end-to-end network paths.

Capacity $C_i$ of a link $i$ at the IP layer can be defined as the maximum possible rate at which data can be transferred at that link. Available bandwidth $A_i$ for the same link $i$ is the average residual capacity of the link over a certain period of time. This is the average capacity available at link $i$ to a new flow transmitting data through it. It is a dynamic measure that varies with the time and depends on the traffic load at that link. As explained in [5], the instantaneous utilization of a link $i$ is either 0 or 1, that is either the link is transmitting data packets at its full capacity $C_i$ or is idle. So the instantaneous available bandwidth $A_i$ is either 0 or $C_i$. However, this instantaneous rate is not of much significance to the data flows, which are more interested in knowing the bandwidth over a given time interval. Thus, the available bandwidth $A_i$ of link $i$ with capacity $C_i$, over a certain period of time can be defined as:

$$A_i = (1 - \rho_i)C_i, \tag{1.1}$$

where $\rho_i$ is the average utilization of the link $i$ during a certain time interval. Fig. 1 (taken from [5]) shows a pipe model of a three-hop network path with fluid traffic,

where the width of each pipe represents the capacity of the link. The average utilization $\rho_i$ of the link $i$ is the width of the shaded part of the pipe, while the unshaded area is the unused capacity $A_i$ of the link $i$. Thus, $C_1$, $C_2$, and $C_3$ represent the capacities of the three links, while $A_1$, $A_2$, and $A_3$ are the available bandwidths.

We extend the definitions of capacity and available bandwidth of individual links to bottleneck and available bandwidth over an end-to-end path as follows.

**Definition 1.** Bottleneck bandwidth is the capacity of the slowest link of an end-to-end path and is the maximum transmission rate that can be achieved between two endpoint hosts in negligible cross-traffic conditions.

**Definition 2.** Available bandwidth is the minimum residual capacity among all the links of an end-to-end path and is the maximum transmission rate at which a flow can send its data.

For an m-hop end-to-end path, bottleneck bandwidth $C$ is given by:

$$C = \min_{i=1,\ldots,m} C_i, \tag{1.2}$$

where $C_i$ is the capacity of the $i^{th}$ link. Such a link with the minimum capacity is also called the *bottleneck link* of the path. Thus, link 1 is the bottleneck link in the three-hop pipe model shown in Fig. 1 and $C_1$ is the bottleneck bandwidth of the path. Similarly, the available bandwidth $A$ of an m-hop path is given as:

$$A = \min_{i=1,\ldots,m} A_i, \tag{1.3}$$

where $A_i$ is the available bandwidth of the $i^{th}$ link. It is highly dynamic as it depends on the amount of cross-traffic present in the links. The link which limits the available bandwidth is also called the *tight link* of the path. Thus, in the pipe model link 3 is the tight link with $A_3$ as the available bandwidth of the path.

## B. Importance in Applications

As suggested in [6] and [7], the performance of several applications could improve from knowing the bottleneck and available bandwidth of an end-to-end path. For instance, present real-time video applications can implement congestion control and dynamically re-scale the enhancement layer of the video stream to any desired bitrate. Knowing the bandwidth would allow them to choose an appropriate encoding scheme and utilize the current network conditions optimally. Network clients or distributed applications, which require service from replicated servers, can use this information to choose the best server or proxy. A server with the highest available bandwidth path might have the shortest response time. Another advantage is to overlay networks that monitor network paths to set up overlay routes. This information would help them to setup better overlay routes and improve end-user QoS.

The most important use of this information is in congestion control employed by various Internet applications and protocols. Most of the congestion control methods work on the principle of AIMD [8]. For example, the TCP congestion control increases the flow transmission rate step-by-step (exponentially during the *slow start* phase and linearly during *congestion avoidance* phase), searching for a higher rate, until the network is congested and the packets are dropped. Then the control-method decreases the flow-rate by a factor of two and again starts probing for a higher rate. The knowledge of available or bottleneck bandwidth can be very useful here. The control-method can use it as the upper bound on the transmission rate and can prevent data loss during the congestion avoidance phase. Many real-time Internet applications like video-streaming, which have rigid time constraints on data-packet arrival, can benefit largely from it. With such a control-method, these applications do not have to lose data packets and wait for retransmissions in the search for a higher

flow-rate.

## C. Research Contribution

Recall that in [9], Kang *et al.* conducted stochastic analysis on network path with a single congested router and proposed a queuing model for an Internet router. This model is further used to construct an asymptotically-accurate bandwidth estimation method under the condition of arbitrary non-fluid cross-traffic. Following up with this work, we propose a recursive extension of the queuing model in [9] to network paths with multiple congested routers. Based on our recursive queuing model, we develop an asymptotically accurate bandwidth estimation method *Envelope*. The key idea of this method is to use two small packets to surround the probing packet-train and preserve the spacing, between the first and the last probing packet of the train, at each router in the path-suffix. Consequently, the receiver can use spacings between the two small packets to recursively estimate bandwidth for each congested router along the path. These small packets are termed as envelope packets.

One measurement using Envelope takes many phases. The number of phases is one less than the number of hops in the end-to-end path. In the first phase, the probing packet-trains are dropped at the second node and in each successive phase, they are dropped at successive nodes along the path. Such a node is termed as sink-node during the phase. In each phase, envelope-packets preserve the average inter-packet spacing of the probing train after the sink-node up to the receiver. Receiver calculates the average inter-packet by alternate sampling of the envelope-packet-pairs and estimates the capacity and available bandwidth of the congested link following the the sink-node. During this process, Envelope also locates the *tight* link, which is the hop with minimum available bandwidth along the path. To our knowledge, this is

the first method that can measure bottleneck and available bandwidth simultaneously in end-to-end Internet path under heavy cross-traffic conditions.

Similar ideas to preserve the spacing between the first and the last probing packet in the train over path-suffix can be found in [10] and Pathneck [11]. However, the three techniques differ in the way the packet-trains are oriented with respect to the small surrounding packets used for spacing preservation. In [10], the authors use cartouche trains, in which the packet-trains are interleaved with spacing-preserving packets, while in Pathneck, there are as many small spacing-preserving packets at each end of the packet-train as the hops. Pathneck locates the tight link and provides an upper bound on the available bandwidth, while the method in [10] estimates the bottleneck bandwidth.

The rest of the thesis is organized as follows. Chapter II reviews the related work done in bandwidth estimation and describes the stochastic queuing model [9]. In chapter III, we verify this model using simulations in NS-2 [12] for single congested node case. In chapter IV, we develop our recursive model and the estimation methodology Envelope, and discuss the simulation results for different possible topologies. Chapter V compares the bandwidth estimation results of our method Envelope with the results of other existing methods such as IGI [13], Spruce [14], Pathload [15], and CapProbe [16] using extensive simulation in NS-2 [12]. In chapter VI, we evaluate our methodology in terms of various probe-traffic parameters, which affect the accuracy of this method such as probe-traffic packet size, packet-train length, and initial inter-packet spacing. Finally in chapter VII, we enhance our methodology Envelope, which simplifies the estimation equations and performs equally well as Envelope.

# CHAPTER II

# BACKGROUND

In this chapter, we first discuss about the basic Packet-pair technique used in various existing bandwidth estimation methods. Then we briefly talk about various bottleneck and available bandwidth estimation methods. Finally, we introduced the stochastic queuing model proposed by Kang *et al.* [9].

## A. Packet-Pair Estimation

In this section, we briefly discuss the basic packet-pair bandwidth estimation technique and the various issues involved with it. The idea of this technique has evolved from the work by Jacobson [8] and Keshav [17]. The technique is based on statistically measuring the dispersion induced between the probing packet-pairs. The measured dispersion is then used in capacity estimation of an end-to-end path. The probe-traffic source and receiver are the end-point hosts of an end-to-end path. The source sends a chain of similar size packet-pairs to the receiver. Each packet-pair consists of two back-to-back packets. These packets get dispersed as they travel through the links towards the receiver. The dispersion between the packets is introduced due to two types of delays. First is the transmission delay of the packets over the link. Second is the queuing delay introduced because of the packets from other flows (cross-traffic) queuing between the two packets. The dispersion of a packet-pair at the receiver is measured as the amount of time between the the last bit of each packet.

Now, if we assume that there are no flows other then the probe-traffic in the links and the cross-traffic induced queuing is negligible, then the dispersion of a packet-pair of size $q$ at a link with capacity $C_k$ is only due to the transmission delay and is given by $\Delta_k = q/C_k$, assuming the packet-pair queue behind each other at router $R_k$. This

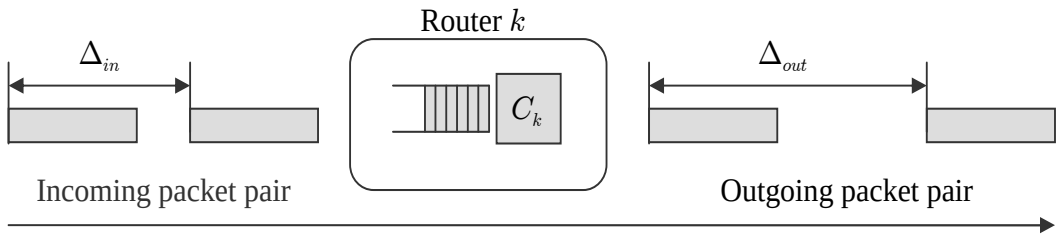



Fig. 2. Dispersion of a packet-pair.

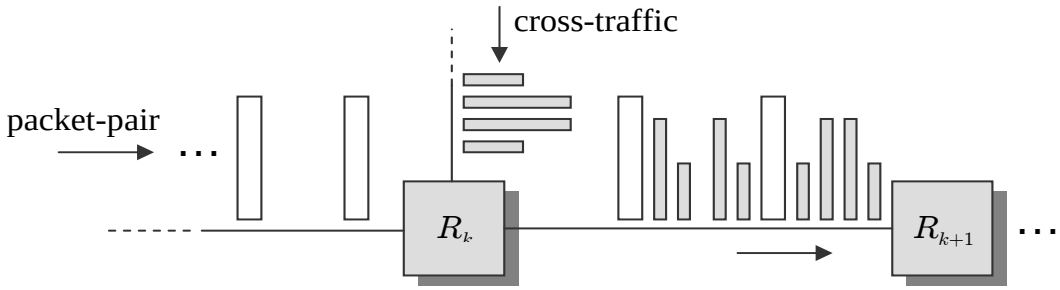


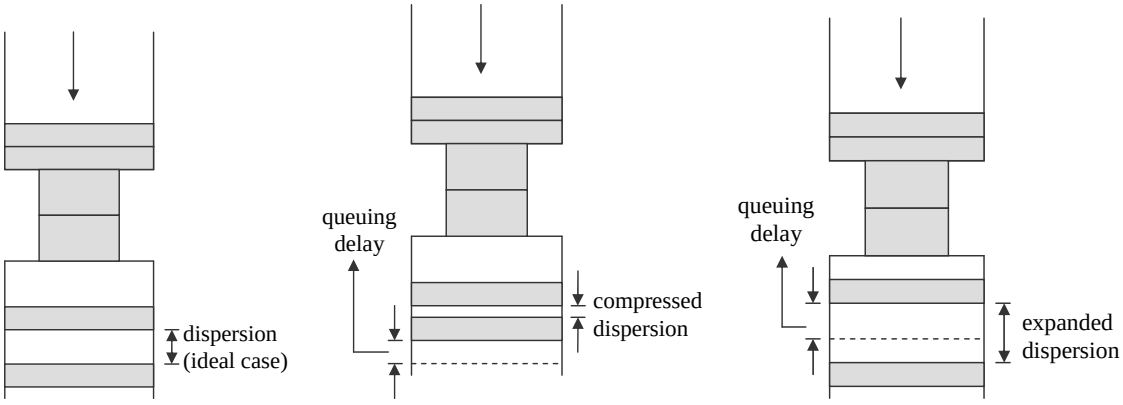

Fig. 3. Cross-traffic packets queuing between packet-pair.

is shown in Fig. 2 [5], where $\Delta_{in}$ and $\Delta_{out}$ is the inter-packet-spacing or dispersion before and after the router $R_k$. If the inter-arrival spacing $\Delta_{in}$ is larger than the transmission delay at any router, then $\Delta_{in} = \Delta_{out}$. Hence, as suggested in [5], the dispersion of a packet-pair after any link of capacity $C_k$ of an end-to-end path is given by:

$$\Delta_{out} = \max(\Delta_{in}, \frac{q}{C_k}). \tag{2.1}$$

Still assuming that the probe-traffic is the only flow over the path, the maximum dispersion between packet-pair would happen in the bottleneck link with minimum capacity. Hence, by measuring the dispersion at the receiver, the bottleneck bandwidth $C$ can be estimated as $C = q/\Delta_{out}$.

However, in the real Internet conditions, the negligible cross-traffic assumption do not hold. As there will always be other flows in the path, along with the probe-traffic, the dispersion of packet-pair is not restricted to the transmission delays. The




(a) Ideal Dispersion (b) Compressed Dispersion (c) Expanded Dispersion

Fig. 4. Noise in the dispersion of packet-pair.

cross-traffic packets might add a random noise between the packet-pair spacing due to queuing delays. Fig. 3 shows the cross-traffic packets queuing between the packet-pair. Both leading and trailing packets of a pair might be queued at a router due to the presence of cross-traffic packets. Thus, either increasing or decreasing the spacing between them, leading to underestimation or overestimation of the path capacity. Fig. 4 shows a pipe model of a path with different dispersions of packet-pairs in the presence of cross-traffic. The second pipe is the bottleneck link that causes the maximum dispersion. The ideal case, where the dispersion is not altered due to the cross-traffic, is shown in Fig. 4(a). If the leading packet of the pair is queued at any router, the dispersion is compressed, as shown in Fig. 4(b). If the trailing packet of a pair is queued at any router, the dispersion is expanded as shown in Fig. 4(c). The challenging task at the receiver is to filter out theses erroneous packet-pair measurements from the ideal ones. Many different statistical schemes have been used in past to do so, such as median or mode based estimation. However, majority of these methods fail to estimate bandwidth with good accuracy in most of the cases [23]. We discuss more about the different methodologies of packet-pair filtering in the next

section.

Packet-train technique is a slight variant of packet-pair technique. Instead of sending two back-to-back packets, source sends $n$ back-to-back packets in a burst, where $n > 2$. The dispersion at the receiver is measured as the time separation between the last bit of the first and the last packets. The measured dispersion is then used to estimate bottleneck bandwidth of the path as:

$$C = \frac{(n-1)q}{\Delta_{rcv}}, \tag{2.2}$$

where $\Delta_{rcv}$ is the measured dispersion at the receiver and $q$ is the packet size of the probe packets. Again, the cross-traffic introduces random noise in the dispersion of packet-train, which needs to be filtered out for good bandwidth estimates.

## B. Related Work

Bandwidth estimation has been extensively researched in the past and is a major focus of current Internet studies [18], [19], [20], [21], [7], [22], [23], [24], [15], [10], [13], [6], [14], [9], [16], [25], [26], [11]. There are two different measures of bandwidth of a link. One is the static measure which is independent of the traffic on the link and is the true capacity of the link. The other is a dynamic measure which depends on cross-traffic, and is the residual capacity of the link. Minimum of the static measure over the links in an end-to-end path is the *bottleneck bandwidth* and the minimum of the dynamic measure is the *available bandwidth* of the path. Link with the minimum available bandwidth is referred to as the *tight* link.

As suggested in [10], these methods can be classified on the basis of various factors. However, here we focus on the one which classifies them according to the metric they measure: bottleneck bandwidth or available bandwidth. Methods [18],

[23], [10], [16], [7], [19] measure the bottleneck bandwidth, while [13], [6], [15], [22], [25], [14], [11] measure the available bandwidth. Most of these bandwidth estimation methods either assume no cross-traffic in their models or assume that the interference due to cross-traffic is non-destructive (zero-mean) during the estimation. However, in the real Internet conditions these assumptions are not valid due to the presence of arbitrary cross-traffic and hence these methods become inaccurate and sometimes converge to wrong estimates.

### 1. Bottleneck Bandwidth Methods

In general, the bottleneck bandwidth estimation techniques rely on the idea of observing transmission delays introduced between two back-to-back packets of a packet-pair or packet-train at the bottleneck link. To filter out the packet-pairs whose spacing is either compressed or extended due to cross-traffic, many of these techniques use *median- or mode-* based filtering, which is based on the assumption that majority of the results are good. As in bprobe [18], one of the first methods to measure bottleneck bandwidth, the authors used packet-pair approach and estimated bandwidth by taking either the intersection or union of predicted intervals. Paxson [19] identified the multimodal nature of the distribution of packet-pair bandwidth measurements and also identified the problem of multi-channel links and acknowledgement compression (TCP) in the sender-based packet-pair estimation. To overcome these problems he proposed the Packet Bunch Modes (PBM) bandwidth estimation technique. PBM sends packet bunch of increasing sizes and uses modal techniques to find multiple modes in the distribution function of the bandwidth estimates. The final estimate of bottleneck bandwidth is made using a complex heuristic which filters out the modes to find a global mode.

In [7], Lai and Baker proposed Potential Bandwidth Filtering (PBF), which is

based on the observation that in the packet-pair bandwidth estimation, a probe-rate (potential bandwidth) less than the bottleneck bandwidth can never predict the correct estimates. Before estimating the bandwidth using relative errors in packet-pair estimation, PBF filters out the packet-pair measurements which predict bottleneck bandwidth equal to the potential bandwidth. Like Paxson [19], Dovrolis *et al.* [23] also highlighted the multimodal nature of bandwidth measurements and contrary to [19] pointed out that the global mode of bandwidth measurements may not always be the correct bandwidth estimate. They also observed that packet-pair is better than the long packet-trains in estimating bandwidth as the interference due to cross-traffic increases with length ($N$) of packet-train. They further observed that variance in measurement decreases with increase in $N$ and for a large value of $N$ the bandwidth measurements become unimodal. They called this mode as Asymptotic Dispersion Rate (ADR) and found out that its value is less than the bottleneck bandwidth. Based on these observations they proposed a bottleneck bandwidth estimation methodology which filters out all, but one, modes in the packet-pair measurements's multimodal distribution, selecting the next highest mode after the ADR. This methodology has been implemented in a tool called pathrate.

Harfoush *et al.* in [10] proposed an end-to-end bandwidth estimation technique over a subpath based upon the idea of packet-pair. The technique uses a large packet to lead the probe packets, so that the probe packets are back-to-back at the first link of the subpath. After the subpath, the inter-packet spacing of the probe-traffic is preserved using a strategy and the bandwidth is calculated. The strategy of preservation is based on the fact that, *if the inter-packet spacing is larger than the maximum transmission delay encountered by packets over a path, then the spacing would be preserved over that path, assuming no cross-traffic.* Kapoor *et al.* in [16] proposed CapProbe, which is based on the idea that if two back-to-back packets are never queued in a

path, then their inter-arrival spacing at the receiver represents the transmission delay in the bottleneck link and this delay would be the minimum among all the packet-pair samples. CapProbe sends a probe of packet-pairs with same packet size and uses a minimum delay condition to reject/select the packet-pair with minimum delay. If all the packet-pair samples are rejected, another probe is sent with different packet size ($\pm 20\%$). This is repeated till one packet-pair is selected as having minimum delay sum.

## 2. Available Bandwidth Methods

In the case of available bandwidth estimation, it is required to have cross-traffic for analysis and most of the analysis assumes it to be generated by a *fluid* process. This is a very strong assumption to make for the cross-traffic and is never true in the real-time analysis.

TOPP [22] is based on a modified packet-pair technique in which the trains of packet-pairs are sent at increasing rates rather then back-to-back. The inter-packet spacing of the packet-pairs at the receiver is averaged over all the pairs of one train to get the output rate of probe-traffic train and these output rates are then used with the input rates to estimate available bandwidth as well as bottleneck bandwidth. Jain *et al.* in [6], [15] proposed an available bandwidth estimation methodology SLoPS, which is based on the observation that if the probe-rate is higher than the available bandwidth, then the inter-packet spacing of probe-traffic at the receiver shows an increasing trend. Using an iterative algorithm, SLoPS converges to a range of available bandwidths. This technique has been used in an available bandwidth estimation tool called Pathload.

In IGI [13], which works on the principle of congesting the link to calculate bandwidth, the initial inter-probe-packet spacing is increased until average sender gap

and average receiver gap are equal. This is the point where the noise in the probe-traffic is zero-mean and probe-rate is equal to the available bandwidth. Another method Spruce [14] is based on the probe gap model (PGM) described in [14] and assumes a single congested node in analysis. Spruce sends the packet-pairs in such a way that it ensures the queue at the bottleneck link does not empty between the departures of the two probe packets in a pair. It then calculates available bandwidth by subtracting the known capacity from the arrival rate at the receiver. The authors have pointed out that Spruce might not perform well in the cases, when the bottleneck link is not the *tight* link or when there are multiple bottleneck links of the same capacity.

Pathneck [11] use packet-trains surrounded by small packets which gets dropped at each hop, called as Recursive Packet Trains (RPT). There are as many small packets at each end of the packet-train as the hops. When the packets are dropped, the router sends two ICMP packets back to the source [27]. At the source, Pathneck use the time gap between the two ICMP packets from each router to estimate the packet-train length on the incoming link of that router. By measuring the changes in the packet-train length, the position of tight link is inferred. Pathneck do not use any model to filter out the noise due to cross-traffic. Hence, it claims to locate the tight link and only provides an upper bound on the available bandwidth.

These issues have been recently highlighted in a study by Kang *et al.* [9]. They proposed a stochastic queuing model for an Internet router using the single-congested-node case and assuming an arbitrary cross-traffic in the congested node. The model estimates the bottleneck and available bandwidth simultaneously with good accuracy for the case with cross-traffic only in bottleneck link (single-congested-node). However, it fails for the multiple congested nodes case, i.e., when there is cross-traffic in multiple links. We will explain this model in detail in the next subsection.

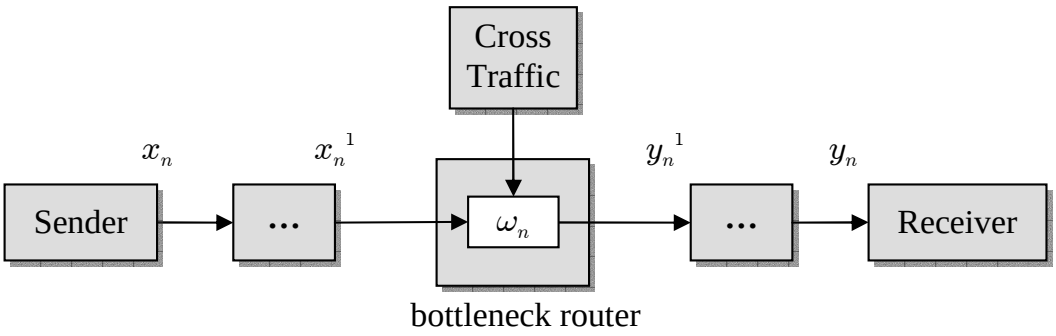



Fig. 5. An end-to-end Internet path.

## C. Stochastic Queuing Model

Here we describe the simple queuing model of a router proposed by Kang *et al.* [9]. This model is based on the queuing theory and the packet-pair bandwidth estimation methodology. It takes into consideration the interference of cross-traffic packets with probe-traffic at the bottleneck router queue. Unlike previous models, this model assumes the cross-traffic to be arbitrary/random not generated by a fluid source. The model analyzes the effect of the destructive-interference by cross-traffic on the spacing of probe-traffic at the receiver and uses it for better packet-pair bandwidth estimation.

The packet-pair bandwidth estimation technique relies on the correct estimation of transmission delay introduced between two back-to-back packets at the bottleneck link. However, due to cross-traffic interference at the routers, random noise (queuing delays) is introduced in addition to the transmission delay between such packets. The queuing model proposed by Kang *et al.* [9], stochastically analyzes this problem and filters out the non-zero-mean noise in the estimation and gives better results in the single-congested-node case. Fig. 5 represents a generic end-to-end internet path, assumed by queuing model, in which routers are present before and after the bottleneck link. This path has only one congested router and no cross-traffic in

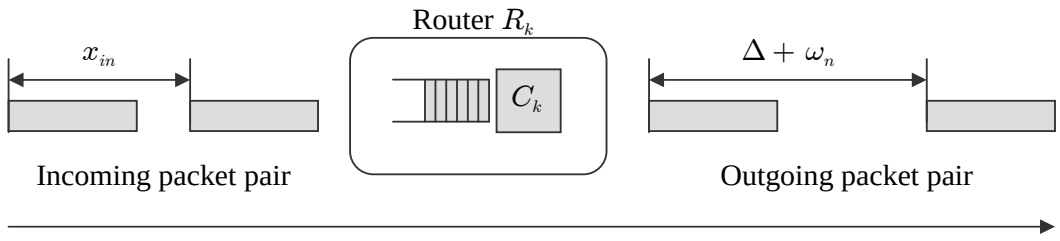



Fig. 6. Packet-pair at the bottleneck router.

the other nodes except the bottleneck. Variable $x_n$ are the inter-departure times of probe-traffic at the sender and $x_n^1$ are the inter-arrival times of probe packets at the bottleneck router. Similarly, $y_n$ and $y_n^1$ are the inter-departure and inter-arrival times of probe-traffic at the bottleneck router and receiver respectively. The noise $\omega_n$ is introduced by the cross-traffic at the bottleneck router. Now we list the theories on which this model is based:

1. Single congested link: pre- and post- bottleneck nodes are not congested and hence $x_n^1$ and $x_n$ are deterministically equal as well as $y_n$ and $y_n^1$.

2. Buffer management schemes do not change the order of packets within each flow and FIFO scheduling is in effect for each flow.

3. Bottleneck node adds a random noise, $\omega_n$, to each received packet, as shown in Fig. 6.

Fig. 7 is taken from [9] and displays a typical queue at the bottleneck router where, $a_{n-1}$, $a_n$, $a_{n+1}$ are the arrival times and $d_{n-1}$, $d_n$, $d_{n+1}$ are the departure times of $(n-1)^{th}$, $n^{th}$, and $(n+1)^{th}$ probe packets respectively. Constant packet size $q$ is used for probe-traffic and $C$ represents the capacity of the bottleneck link. The transmission delay of each probe packet is $q/C = \Delta$. With these theories, departure

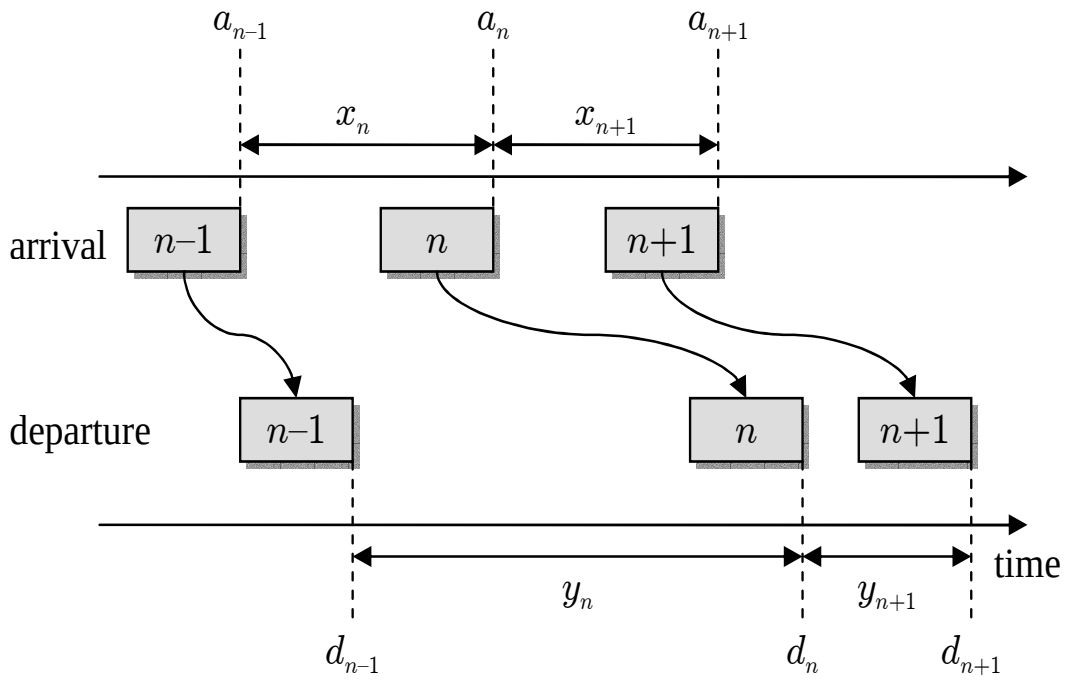



Fig. 7. Queue at the bottleneck router.

times $d_n$ of probe packets at the bottleneck link can be expressed as shown in [9]:

$$d_n = \begin{cases} a_1 + \omega_1 + \Delta, & \text{if } n = 1 \\ \max(a_n, d_{n-1}) + \omega_n + \Delta, & \text{if } n \geq 2 \end{cases}. \tag{2.3}$$

In this formula, $a_n$ is the arrival time of packet $n$. The departure time $d_n$ of a packet not only depends on the arrival time, but also on the number of cross-traffic packets queued in front of it and the departure time of the previous packet $n-1$. Here, if the packet-pair queue behind each other at bottleneck router ($x_n$ is small), then (2.3) is simplified to the following model:

$$d_n = \begin{cases} a_1 + \omega_1 + \Delta, & \text{if } n = 1 \\ d_{n-1} + \omega_n + \Delta, & \text{if } n \geq 2 \end{cases}. \tag{2.4}$$

Hence, the inter-departure delays are given by $y_n = d_n - d_{n-1}$. Define $W_n$ to be the

average of $n$ samples of $y_i$:

$$W_n = \frac{1}{n}\sum_{i=1}^{n} y_i. \tag{2.5}$$

The model (2.4) assumes that the cross-traffic arrival process $r(t)$ , has a finite time average $\bar{r}$:

$$\bar{r} = \lim_{n\to\infty} \frac{1}{t}\int_0^t r(u)\,du < \infty. \tag{2.6}$$

If we sample $r(t)$ using a Poisson sequence of probes at times $t_1$, $t_2$, $\cdots$, and average instantaneous $r(t_i)$, we get (from the PASTA principle [28]):

$$\lim_{n\to\infty} \frac{r(t_1) + r(t_2) + ... + r(t_n)}{n} = \lim_{t\to\infty} \frac{1}{t}\int_0^t r(u)\,du = \bar{r}, \tag{2.7}$$

as long as delays $\tau_i = t_i - t_{i-1}$ are $i.i.d.$ exponential random variables.

Further solving (2.5) with model (2.4), taking $r(t)$ and into account, and fixing $x_i = x$, we get:

$$\lim_{n\to\infty} W_n = \Delta + \frac{x\bar{r}}{C}. \tag{2.8}$$

This is a linear equation in $x$. By using two simulations with signals $x_n = a$ and $x_n = b$ we can get the mean of the measured signals at the receiver as $W_n^a$ and $W_n^b$. Now this is sufficient information to calculate the unknown constants in (2.8):

$$\lim_{n\to\infty} \tilde{\Delta}_n = \lim_{n\to\infty} \left( W_n^a - \frac{W_n^a - W_n^b}{a-b} a \right) = \Delta. \tag{2.9}$$

$$\lim_{n\to\infty} \tilde{C}_n = \lim_{n\to\infty} \frac{q}{\tilde{\Delta}_n} = C. \tag{2.10}$$

$$\lim_{n\to\infty} \left( \frac{W_n^a - W_n^b}{a-b} \tilde{C}_n \right) = \bar{r}. \tag{2.11}$$

Now available bandwidth $A = C - \bar{r}$, details of these derivations can be found in [9].

# CHAPTER III

## SINGLE CONGESTED NODE

In this chapter, we first verify the correctness of the stochastic model (2.3) in estimating bottleneck and available bandwidth for a single congested node case. Then, we compare its results with the other existing methods for bandwidth estimation such as IGI [13], Spruce [14] and Pathload [15]. Finally, we test this model for a path with multiple congested nodes, where we find out that it fails when we have cross-traffic in pre- and post- bottleneck links. All the simulations are done in NS-2 using CBR and TCP cross-traffic in congested links.

### A. Network Topology

As shown in Fig. 8, the network consists of four nodes serving as routers ($R_1$, $R_2$, $R_3$, $R_4$), two nodes (PT) which send and receive the probe-traffic, and six collection of nodes for sending and receiving various kinds of cross-traffic ($CT_1$, $CT_2$ and $CT_3$). The associated links have varied capacities and propagation delays. Some of these parameters are changed as per the requirements of simulation. Capacities $C_1$, $C_2$ and $C_3$ represent base bandwidths of the links between the four routers. The link between routers $R_2$ and $R_3$ is the bottleneck link and has capacity $C_2 = 1.5$ Mb/s and 20 ms propagation delay. The links which are feeding probe- or cross- traffic into the routers are 100 Mb/s with delay 5 ms. All the other remaining links are 10 Mb/s with a 10 ms propagation delay, unless mentioned otherwise. The source of probe-traffic is node $PT_S$ and the probe packets are targeted towards the node $PT_R$ which acts as receiver. It samples the probe packets and compute average inter-packet time to be used in estimating the bottleneck and available bandwidth. Collection of nodes, $CT_2$, is used to send CBR or TCP cross-traffic at the rate $\bar{r}_2$ in the bottleneck link.

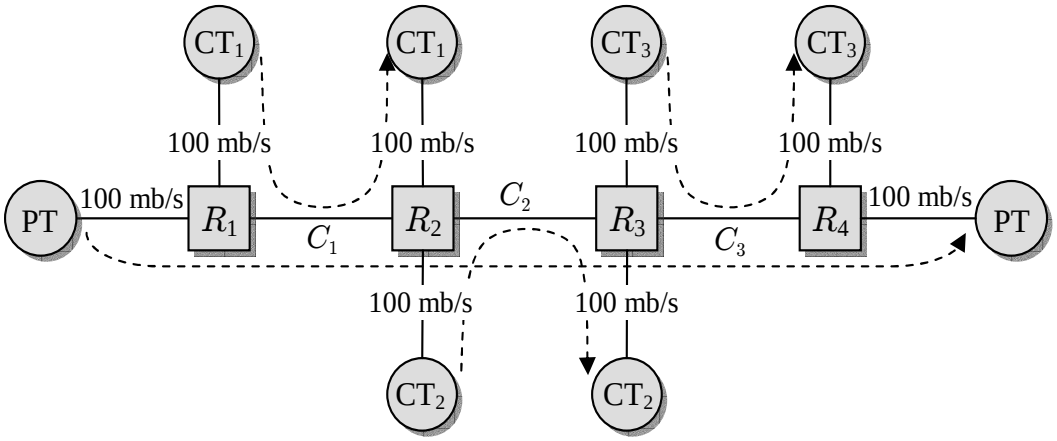



Fig. 8. Arrival and departure of the probe packets at the bottleneck router

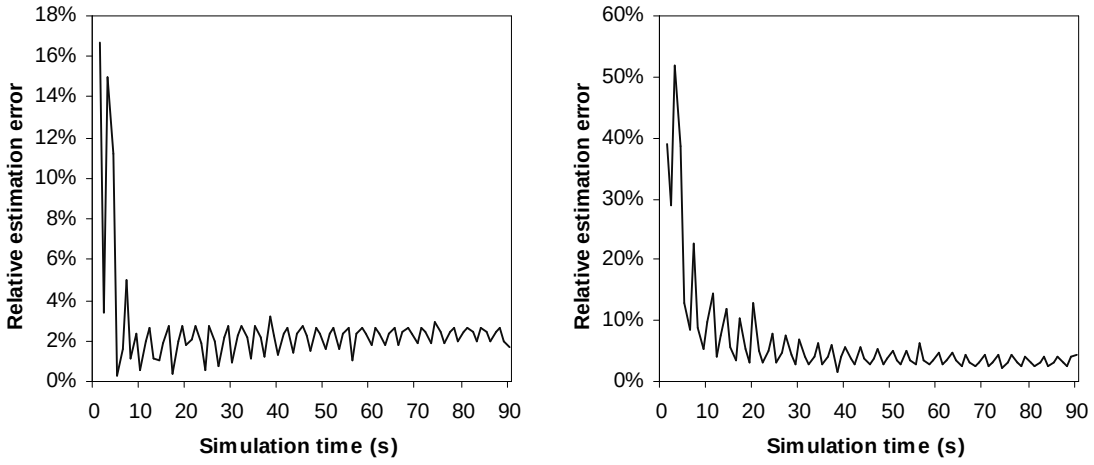
Similarly, $CT_1$ and $CT_3$ are used for the pre- and post- bottleneck cross-traffic with sending rate $\bar{r}_1$ and $\bar{r}_2$ respectively.

## B. Simulations

Here, at first we define the term *relative estimation error* $\tilde{e}$, which will be used in estimating the accuracy of the methods. The same is defined as $\tilde{e} = abs(x - x_{est})/x$, where $x$ and $x_{est}$ are actual and estimated values respectively. Hence, the relative estimation error in bottleneck bandwidth estimation is $\tilde{e}_C = abs(C - \tilde{C})/C$, where, $C$ is the actual capacity and $\tilde{C}$ is the estimated value of capacity. Same in the case of available bandwidth $\tilde{e}_A = abs(A - \tilde{A})/A$. All the simulations are done in NS-2 with CBR or TCP cross-traffic heavily congesting the bottleneck node $R_2$. There is no cross-traffic in pre- and post- bottleneck links $\bar{r}_1 = 0$ and $\bar{r}_2 = 0$.

### 1. CBR Cross-Traffic

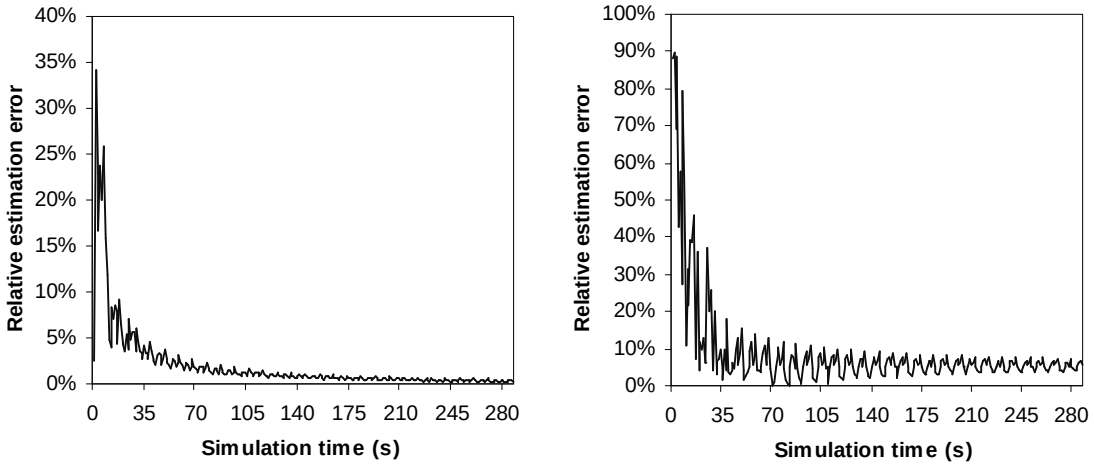
The collection of nodes $CT_2$ are initialized to several UDP sources which sends CBR cross-traffic of 500-byte packets at an average rate of 1.2 Mb/s, thus congesting the




(a) $\tilde{e}_C$ (b) $\tilde{e}_A$

Fig. 9. Relative estimation errors in bandwidth estimation with CBR cross-traffic in single congested bottleneck node.

bottleneck link by 80%. The probe-traffic sender PT sends packet-pairs of 1,500 bytes with alternate intra-packet-pair spacing $x_n^a$ and $x_n^b$. The inter-packet-pair spacing is derived from an exponential distribution in order to have poisson sampling at the receiver. The probe-traffic is sent at an average rate of 75 Kb/s (5% of bottleneck capacity). The probe-traffic receiver PT computes the average intra-packet-pair spacing $W_n^a$ and $W_n^b$ by alternate sampling of the packet-pairs at the receiver. Using $W_n^a$ and $W_n^b$ with (2.9), (2.10), and (2.11) provide accurate results for bottleneck and available bandwidth with relative estimation errors $\tilde{e}_C$ and $\tilde{e}_A$ less than 5%. Fig. 9 shows $\tilde{e}_C$ and $\tilde{e}_A$ for a 90 second simulation. Within the first 15 seconds, the error $\tilde{e}_A$ becomes less than 10% and gradually converges below 5%. In the case of $\tilde{e}_A$, the convergence is even faster.

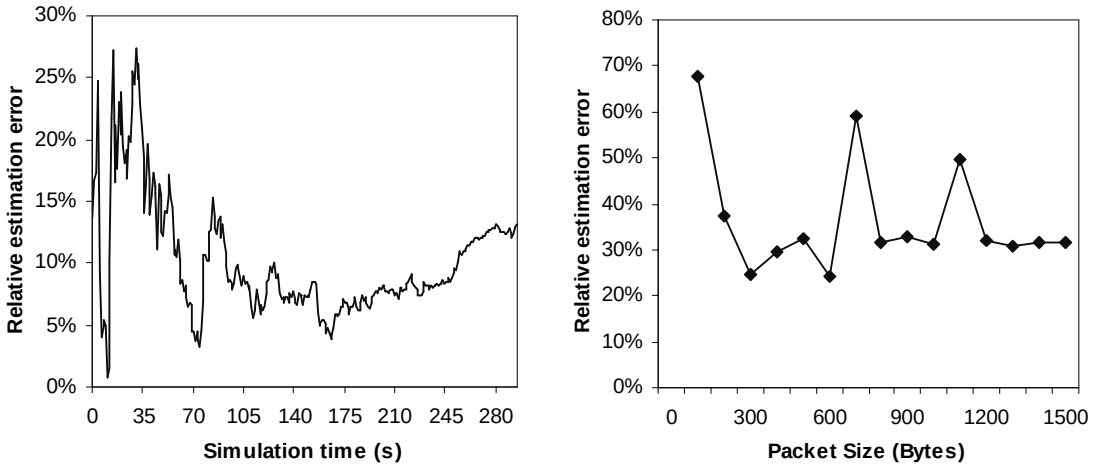



(a) $\tilde{e}_C$ (b) $\tilde{e}_A$

Fig. 10. Relative estimation errors in bandwidth estimation with TCP cross-traffic in single congested bottleneck node.

### 2. TCP Cross-Traffic

Here, cross-traffic nodes $CT_2$ are attached to four FTP sources which generates TCP cross-traffic over the bottleneck link at an average rate of 1.31 Mb/s. Thus, the utilization is 87%. The TCP cross-traffic consists of different packet sizes of 600, 800, 1000 and 1200 bytes. The probe-traffic sender PT, sends packet-pairs of 1,500 bytes at an average rate of 75 Kb/s, the same way as in the CBR case. The initial intra-packet-pair spacing is $x_n^a = 7.5$ ms and $x_n^b = 3.5$ ms. After 270 seconds of simulation, the average intra-packet-pair spacing computed at the PT receiver is $W_n^a = 11.06$ and $W_n^b = 14.59$. These $W_n^a$ and $W_n^b$ together with (2.9), (2.10), and (2.11) estimates bottleneck bandwidth $\tilde{C} = 1503.37$ ($\tilde{e}_C = 0.22\%$) and available bandwidth $\tilde{C} = 179.39$ ($\tilde{e}_A = 5.58\%$). The relative estimation errors $\tilde{e}_C$ and $\tilde{e}_A$ are shown in Fig. 10. The trend in convergence is very similar to the case with CBR cross-traffic. After the first 35 seconds of simulation, the available bandwidth estimate is within 10% of $A$. These results validate the asymptotic behavior of the estimation methodology,




(a) Spruce (b) IGI

Fig. 11. Relative estimation errors in available bandwidth estimation in (a) Spruce (b) IGI with different probe-packet size.

using queuing model (2.3) for a single congested node.

## C. Comparison

Now, we compare the queuing model (2.3) results with the results of other bandwidth estimation methods, such as, IGI [13], Spruce [14], Pathload [15], and CapProbe [16], through NS-2 simulations. We use the same topology as shown in Fig. 8 with CBR and TCP cross-traffic in the bottleneck node $R_2$ as discussed in the previous subsection (utilization: 80-90%). Both Spruce and IGI use the known bottleneck bandwidth $C_1$ for their estimations of available bandwidth $\tilde{A}$. Queuing model (2.3) computes $\tilde{C}_1$ and use it in the estimation of $\tilde{A}$. Pathload estimates a range of available bandwidth without any knowledge of $\tilde{C}_1$. For our analysis, we took the mean of the range of the estimates produced by Pathload.

Fig. 11 shows the convergence of bandwidth estimation error in Spruce with simulation time and the same in IGI with different probe-packet size. The comparison

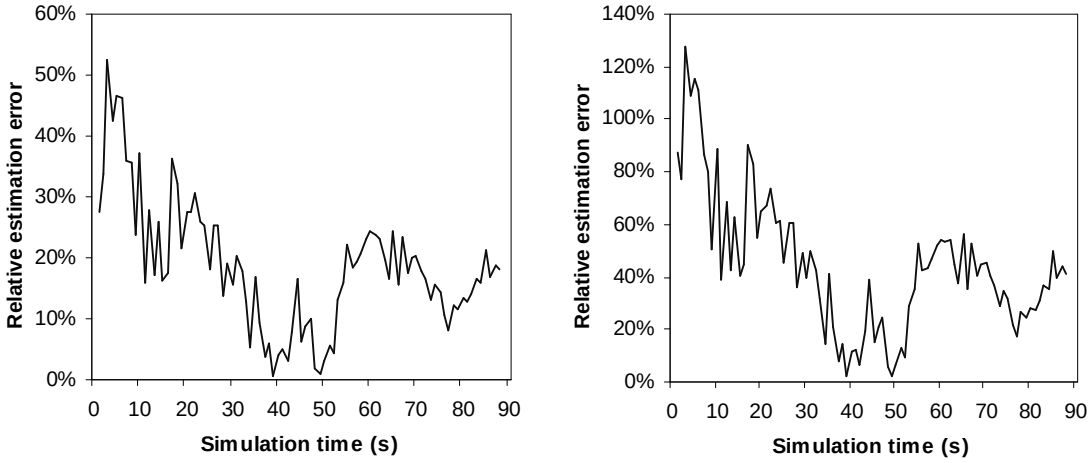
Table I. Different Available Bandwidth Estimation Methods

| Cross-Traffic Source | Relative estimation error $\tilde{e}_A$. | | | |
|---|---|---|---|---|
| | Model(2) | Pathload | Spruce | IGI |
| CBR | 4.01% | 20.00% | 0.01% | 24.17% |
| TCP | 5.58% | 25.00% | 13.37% | 34.00% |

of relative estimation error $\tilde{e}_A$ in different methods is shown in Table I. With CBR cross-traffic, Spruce performs even better than the queuing model (2.3) while Pathload and IGI have very high errors. However, with random cross-traffic TCP, queuing model (2.3) performs almost 2.5 times better than the next best method Spruce and almost 5-7 times better than Pathload and IGI.

## D. Multi-Congested Case Failure

Kang *et al.* in [9] suggested, that the queuing model (2.3) fails in the case of multiple congested nodes case. Therefore, we tested the model (2.3) for the topology shown in Fig. 8, with $C_1 = 1.5$ Mb/s, $C_2 = 2$ Mb/s and $C_3 = 2$ Mb/s. The cross-traffic in the bottleneck node $R_2$ is $\bar{r_2} = 1$ Mb/s (utilization of 67%) and the pre- and post- bottleneck cross-traffic $\bar{r}_1$ and $\bar{r}_2$ are also non-zero (70-80% utilization). The relative estimation errors $\tilde{e}_C$ and $\tilde{e}_A$, shown in Fig. 12, confirm that the queuing model (2.3) does not converge to good estimates if there is cross-traffic in the nodes other than bottleneck. If we compare the errors in Fig. 12(a) and 12(b), interestingly both $\tilde{e}_C$ and $\tilde{e}_A$ follow the same convergence. In the 40-50 seconds section they both converge to good estimates. This proves that the queuing model (2.3) behaves well with multiple congested nodes. However, the convergence is not asymptotic. Rather, it is random. Hence, this model can not be used as it is to construct bandwidth




(a) $\tilde{e}_C$ (b) $\tilde{e}_A$

Fig. 12. Relative estimation errors $\tilde{e}_C$, $\tilde{e}_A$ with $C = 1.5$ Mb/s and 67% utilization when there is CBR cross-traffic in all three routers $R_1$, $R_2$, $R_3$.

estimators for multiple congested nodes case. In the next section, we build upon this model and extend it to multiple case.

# CHAPTER IV

# MULTIPLE CONGESTED NODES

In this chapter, we extend the queuing model (2.3) proposed in [9] to multiple congested nodes and develop a methodology Envelope to estimate bottleneck and available bandwidth over such a path using the extended model.

## A. Recursive Model

Stochastic queuing model proposed by Kang *et al.* in [9] can be recursively extended to the multi-hop case. This is because the inter-packet spacing $x_n^{(k)}$ of the probing packet-pairs arriving at the router $R_k$ is also the inter-packet spacing of the same probing packet-pairs departing from the previous router $R_{k-1}$, i.e.,

$$x_n^{(k)} = y_n^{(k-1)}. \tag{4.1}$$

Here, we assume that the inter-packet spacing $x_n^{(k)}$ of the probe-traffic always remains less (in statistical sense) than the queuing delays experienced by the leading probe packet of the packet-pair at each router. Hence, the probe packets always queue behind each other at each router of an end-to-end path. This can be maintained by choosing appropriate initial inter-packet spacing $x_n^{(0)}$. The following lemma states the condition which assures that the probe-packets of packet-pair always queue behind each other at each router.

**Lemma 1.** *Given a network path of m routers $R_1$, $R_2$, ... $R_m$ interconnected by links with bandwidths $C_1$, $C_2$, ... $C_m$ respectively, a sufficient condition for the two packets $P_{2i}$ and $P_{2i-1}$ in the $i^{th}$ probing-pair to queue behind each other at $R_k$ is*

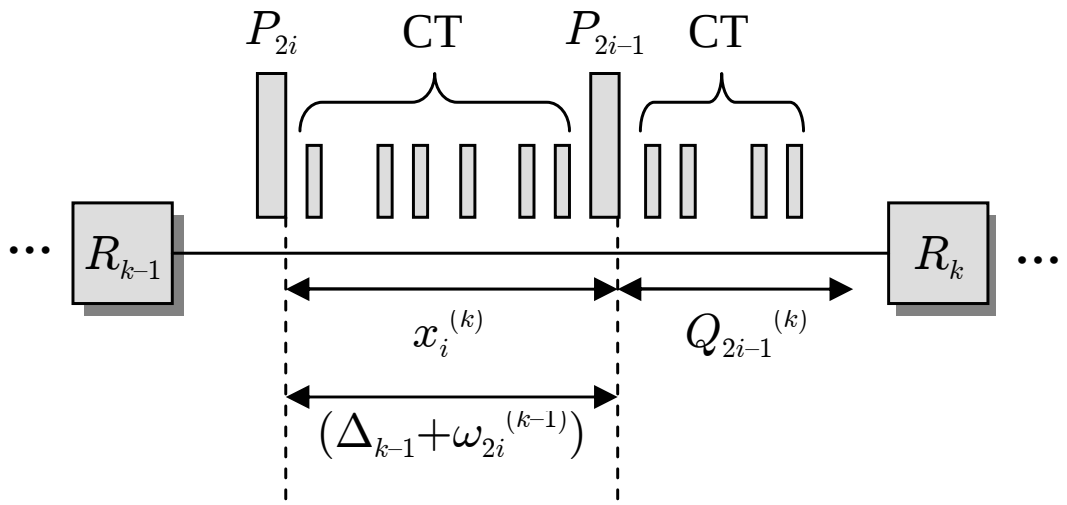



Fig. 13. Cross-Traffic (CT) interferes with probe packets at routers $R_{k-1}$ and $R_k$.

$$\frac{q}{C_{k-1}} + \omega_{2i}^{(k-1)} \leq \frac{q}{C_k} + Q_{2i-1}^{(k)}, \tag{4.2}$$

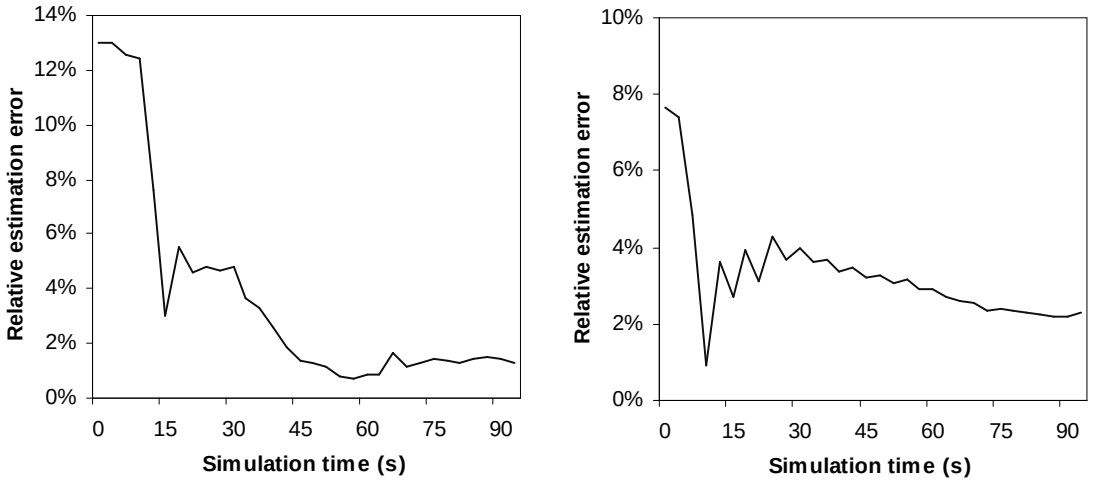
where $q$ is the probing packet size, $\omega_{2i}^{(k-1)}$ is the random noise introduced to the spacing between $P_{2i}$ and $P_{2i-1}$ by the cross-traffic at router $R_{k-1}$, and $Q_{2i-1}^{(k)}$ is the queuing delay the packet $P_{2i-1}$ experienced at router $R_k$, as shown in Fig. 13.

An important corollary of Lemma 1 is that if the router $R_k$ is not congested (i.e., $Q_{2i-1}^{(k)} \approx 0$), then the lemma is simplified to the PBF [7] condition in packet-pair bandwidth estimation. That is, to estimate bottleneck capacity, the probe-rate should be higher than the bottleneck capacity.

Combining Lemma 1 with the queuing model (2.3), we have:

$$y_i^{(k)} = \Delta_k + \omega_{2i}^{(k)}. \tag{4.3}$$

This leads to the following recursive queuing model, given the instantaneous cross-




(a) $\tilde{e}_C$, $C_1 = 1.5$ Mb/s (b) $\tilde{e}_C$, $C_2 = 1.5$ Mb/s

Fig. 14. Relative estimation errors $\tilde{e}_C$ with topology having two-congested nodes.

traffic rate $r_k(t_i)$ and small spacing $x_i^{(k)}$ at router $R_k$:

$$y_i^{(k)} \approx \Delta_k + \frac{r_k(t_i)x_i^{(k)}}{C_k} = \Delta_k + \frac{r_k(t_i)y_i^{(k-1)}}{C_k}. \tag{4.4}$$

Notice that if the inter-packet spacing $y_n^{(k-1)}$ and $y_n^{(k)}$ of the probing pairs departing from router $R_{k-1}$ and $R_k$ is known, then we can use (4.4) along with (2.9), (2.10), and (2.11) to obtain bandwidth estimation for $R_k$ using the same method as is used in [9] for single congested node paths. To verify the recursive model (4.4), we did a simulation in NS-2 with a topology having two consecutive congested nodes $R_2$ and $R_3$ as shown in Fig. 8. We found, that if the values of $x_n^{(k)}$ and $y_n^{(k)}$ are *accurately* known, (4.4) leads to good bandwidth estimation. In NS-2, it is possible to sample the probe signal at the congested nodes and we can find the exact inter-packet spacing ($x_n^{(k)}$, $y_n^{(k)}$) of probe packets before and after the congested nodes. When these values are used with the model (4.4), we get accurate results with less then 3% relative estimation error $\tilde{e}_C$, as shown in Fig. 14. The results verify that the queuing model can be applied recursively to multiple congested nodes if we have approximate

information of the inter-packet spacing of probe-traffic at each node.

To obtain departing spacing information at router $R_k$, $y_n^{(k)}$ need to be preserved along the path-suffix up to the receiver. We formally state a sufficient condition for spacing preservation using the following lemma. The lemma assumes that the queue at the routers drain between the arrival of the two packets of a packet-pair. Hence, the leading packet of a packet-pair does not introduce any residual noise to the queuing delay of the trailing packet. However, the assumption requires the inter-packet spacing to be very large so that the probability of a queue getting drained is high.

**Lemma 2.** *Given a network path of m routers $R_1$, $R_2$, ... $R_m$ interconnected by links with bandwidths $C_1$, $C_2$, ... $C_m$ respectively, a sufficient condition for the mean of the departing spacing $E[y_i^{(k)}]$ at $R_k$ to be preserved along the path-suffix from $R_{k+1}$ up to receiver is:*

$$\frac{q}{C_k} + \omega_{2i}^{(k)} > \max_{k<j\leq m} \left( \frac{q}{C_j} + Q_{2i-1}^{(j)} \right), \tag{4.5}$$

where $q$ is the probing packet size, $\omega_{2i}^{(k)}$ is the random noise introduced to the probing spacing by the cross-traffic at router $R_k$, and $Q_{2i-1}^{(j)}$ is the queuing delay packet $P_{2i-1}$ experienced at router $R_j$.

If there is no cross-traffic in the path-suffix from router $R_{k+1}$ ($Q_{2i-1}^{(j)} \approx 0$, $k < j \leq m$), then (4.5) is simplified to the packet-pair spacing preservation condition stated in [10].

Note that Lemma 2 contradicts Lemma 1. In the next section, we develop a methodology *Envelope* that preserves $E[y_i^{(k)}]$ based on Lemma 2, but at the same time also satisfies the condition in Lemma 1. This methodology allows us to use the recursive queuing model (4.4) to estimate bandwidths over paths with multi-congested

nodes.

## B. Envelope

Now, we know that the recursive queuing model (4.4) for multiple congested nodes works correctly, given the knowledge of the inter-packet spacing of probe-traffic over the congested nodes. However, in the real Internet conditions, it is not possible for the receiver to know the inter-packet spacing $y_n^{(k)}$ of probe-traffic at each router due to the cross-traffic in the path-suffix. This problem is our focus here. In this section, we develop a method *Envelope* that measures such spacing at each router and use it in the recursive model to obtain accurate bandwidth estimation.

As we know, the cross-traffic in congested or non-congested nodes in the path-suffix (nodes after the congested nodes we are presently studying), might change the inter-packet spacing of probe-traffic from the time it departs from the congested node and arrives at the receiver node, where it will be sampled. Thus, in order to obtain an estimate of the inter-packet spacing $x_n^{(k)}$ and $y_n^{(k)} = x_n^{(k+1)}$, we need to preserve the spacing over the path-suffix nodes until the probe-traffic reaches the receiver. As shown in [9], to preserve the mean of the inter-packet spacing $E[y_n^{(k)}]$ over a node, the noise introduced by the cross-traffic at the node is required to be zero-mean. This is possible if the spacing $y_n^{(k)}$ is large enough, as suggested in Lemma 2. However, if the probe-traffic has such large inter-packet spacing, our assumption for the recursive model (4.4) stated in Lemma 1 is contradicted.

Therefore, to obtain departing probe spacing information at intermediate routers, instead of using packet pairs, we use a probing-train of $n$ packets and only preserve the spacing between the first and the last packet in the train over the path-suffix in consideration. This is achieved using enveloped packet-trains, in which probing-

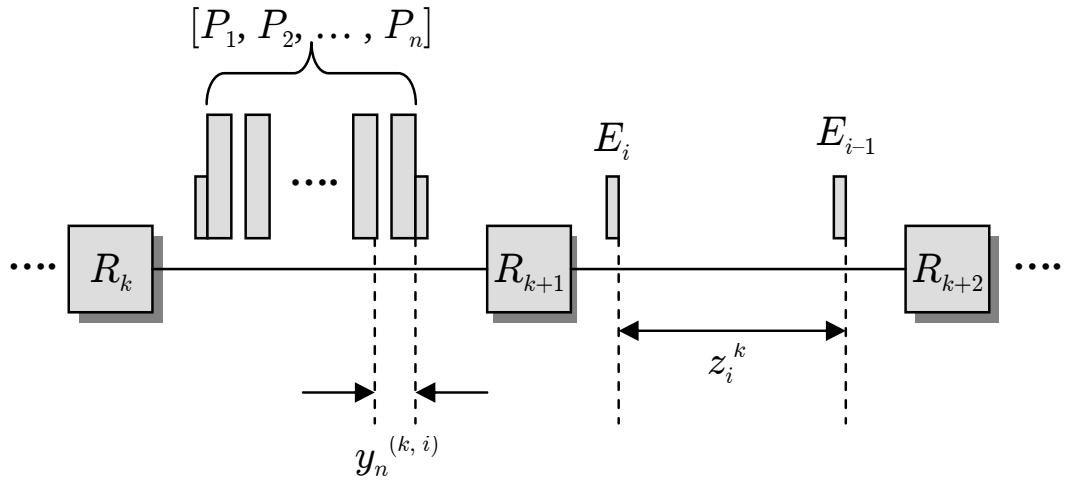



Fig. 15. A probe-burst $[P_1, P_2, ..., P_n]$ of $n$ packets enveloped by two small packets $[E_i, E_{i-1}]$ at router $R_k$.

trains of large packets are surrounded by two small envelope-packets. This is shown in Fig. 15, where a probe-traffic burst $[P_1, P_2, ..., P_n]$ of $n$ packets is enveloped by two small envelope-packets $[E_i, E_{i-1}]$ at router $R_k$. The probe-traffic is dropped at $R_{k+1}$ and the envelope-packets preserve probe-traffic inter-packet spacing until the receiver is reached. The following lemma expresses the condition which assures that the inter-envelope-packet spacing $z_i^{(k)}$ is preserved in the path-suffix after router $R_k$.

**Lemma 3.** *Given a network path of m routers $R_1$, $R_2$, ... $R_m$ interconnected by links with bandwidths $C_1$, $C_2$, ... $C_m$ respectively, a sufficient condition for the inter-packet spacing $z_i^{(k)}$ between envelope pairs to be preserved after router $R_k$ is*

$$\frac{nq_p}{C_k} + \omega_i^{(k)} > \max_{k<j\leq m} \left( \frac{q_e}{C_j} + Q_{i-1}^{(j)} \right), \tag{4.6}$$

where $q_p$ and $q_e$ are the packet size for probing packets and envelope packets respectively, $\omega_i^{(k)}$ is the total amount of random noise introduced to the inter-packet spacings of the probing train at router $R_k$, and $Q_{i-1}^{(j)}$ is the queuing delay the preceding envelope-packet experienced at any router $R_j$ in the path-suffix.

To satisfy the condition in lemma 3, the number of packets $n$ in the probe-burst or packet-train has to be large. Even a larger packet size of the probe-packets should also help with the purpose. By limiting the TTL values, we force the probing packets to be dropped at $R_{k+1}$, while the envelope packets $E_i, E_{i-1}$ are further sent towards the receiver. With the probe-train being dropped at router $R_k$, the envelope packets $E_i, E_{i-1}$ are sampled by the receiver and the inter-envelope-packet spacing $z_i^{(k)}$ is calculated simultaneously. The spacing $z_i^{(k)}$ reflects the collective inter-packet spacings of one probe-traffic burst:

$$z_i^{(k)} = (n-1)\tilde{y}_i^{(k)} \approx (n-1)\bar{y}_i^{(k)} + \Delta_k, \tag{4.7}$$

where $\tilde{y}_i^{(k}$ and $\bar{y}_i^{(k)}$ are respectively the average sampled and actual inter-departure dispersion between the packets in the $i^{th}$ train at router $R_k$. Now combining the recursive model (4.4) and (4.7), we obtain:

$$\tilde{y}_i^{(k)} = \frac{z_i^{(k)}}{n-1} = \bar{y}_i^{(k)} + \frac{\Delta_k}{n-1} = \frac{n}{n-1}\Delta_k + \frac{r_k(t_i)\bar{y}_i^{(k-1)}}{C_k}. \tag{4.8}$$

Define $E_l^{(k)}$ to be the average of $l$ samples of $\tilde{y}_i^{(k)}$:

$$E_l^{(k)} = \frac{1}{l}\sum_{i=1}^{l}\tilde{y}_i^{(k)} = \frac{1}{l}\sum_{i=1}^{l}\frac{z_i^{(k)}}{n-1}. \tag{4.9}$$

Finally, taking limits in (4.8) and (4.9), we get:

$$\lim_{l\to\infty} E_l^{(k)} = \frac{n}{n-1}\Delta_k + \frac{E[\bar{y}_i^{(k-1)}]\bar{r}_k}{C_k}. \tag{4.10}$$

Once the service time of probing packet $\Delta_{k-1}$ at router $R_{k-1}$ is known from the previous phase of the measurement, $E[\bar{y}_i^{(k-1)}]$ can be obtained from $E[\tilde{y}_i^{(k-1)}]$ using (4.8). Hence, there are two unknown constants in (4.10), the transmission delay $\Delta_k$ of probe packets at router $k$, and the utilization factor $\rho_k = \frac{\bar{r}_k}{C_k}$ of router $k$. These can be

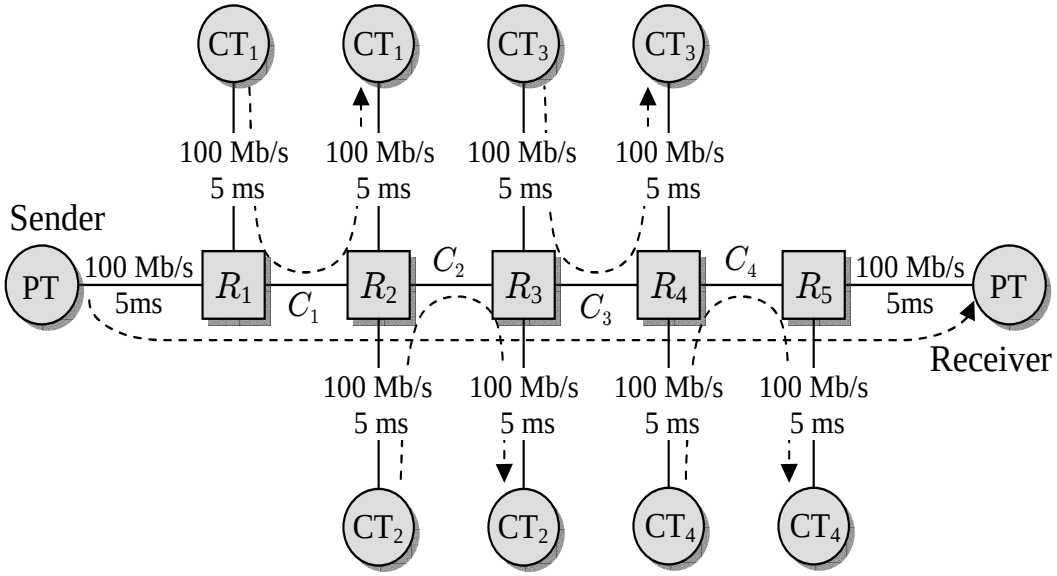



Fig. 16. Simulation topology for multiple congested routers case.

obtained by using measurement run with two alternate packet-trains with different initial inter-packet spacing $x^{(0)} = a$ and $x^{(0)} = b$, and computing metrics $E_l^{(k,a)}$ and $E_l^{(k,b)}$ at the receiver by alternate sampling of envelope-packets. The following equations provide asymptotic convergence of $\Delta_k$ and $\rho_k$:

$$\lim_{l\to\infty} \tilde{\Delta}_k^l = \frac{n-1}{n} \lim_{l\to\infty} \frac{E_l^{(k-1,a)} E_l^{(k,b)} - E_l^{(k-1,b)} E_l^{(k,a)}}{E_l^{(k-1,a)} - E_l^{(k-1,b)}} = \Delta_k. \tag{4.11}$$

$$\lim_{l\to\infty} \tilde{\rho}_k^l = \lim_{l\to\infty} \frac{E_l^{(k,a)} - E_l^{(k,b)}}{E_l^{(k-1,a)} - E_l^{(k-1,b)}} = \rho_k. \tag{4.12}$$

Using (4.11) and (4.12), capacity $C_k$ and available bandwidth $A_k$ of the link $k$ can be estimated as:

$$\lim_{l\to\infty} \tilde{C}_k^l = \lim_{l\to\infty} \frac{q}{\tilde{\Delta}_k^l} = C_k. \tag{4.13}$$

$$\lim_{l\to\infty} \tilde{A}_k^l = \lim_{l\to\infty} \left(1 - \tilde{\rho}_k^l\right) \tilde{C}_n = A_k. \tag{4.14}$$

Table II. Capacities and Available Bandwidths During Simulations.

| Different Cases | Different link bandwidths (Mb/s) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $C_1$ | $A_1$ | $C_2$ | $A_2$ | $C_3$ | $A_3$ | $C_4$ | $A_4$ |
| Case-I | 2 | 0.4 | 1.5 | 0.3 | 0.8 | 0.16 | 1.5 | 0.3 |
| Case-II | 2 | 0.4 | 1.5 | 0.25 | 0.8 | 0.4 | 1.5 | 0.35 |
| Case-III | 20 | 4 | 15 | 3 | 8 | 1.6 | 15 | 3 |
| Case-IV | 20 | 4 | 15 | 2.5 | 8 | 4 | 15 | 3.5 |
| Case-V | 2 | 0.4 | 0.8 | 0.4 | 1.5 | 1.25 | 2 | 1.6 |

## C. Simulations

In this section, we verify the new methodology Envelope through NS-2 simulations. The network topology we use is given in Fig. 16. The network consists of five nodes serving as routers $R_1$, $R_2$, $R_3$, $R_4$, and $R_5$, two nodes PT which send and receive the probe-traffic, and eight collection of nodes $\mathrm{CT}_1$, $\mathrm{CT}_2$, $\mathrm{CT}_3$, and $\mathrm{CT}_4$, which send and receive CBR or TCP cross-traffic at rates $\bar{r}_1$, $\bar{r}_2$, $\bar{r}_3$, and $\bar{r}_4$ respectively. The associated links have capacities and propagation delays as indicated in the Fig. 16. Capacities $C_1$, $C_2$, $C_3$, and $C_4$ represent base bandwidths of the links between the five routers.

We report experimental results in five different network settings as showed in Table II. We use CBR cross-traffic for the first four cases and use TCP cross-traffic for all five cases. Table II also lists the capacity and available bandwidth of each link, where the shaded values are bottleneck capacities and available bandwidths. Note that in cases I and III, the bottleneck link is the tight link; in cases II and IV, the bottleneck link is *after* the tight link; while in case V, the bottleneck link is *before* the tight link.

In all four cases, CBR cross-traffic is generated by several UDP sources attached to the collection of nodes $CT_1$, $CT_2$, $CT_3$, and $CT_4$. These sources send CBR cross-traffic of 500-byte packets at an average rate of 80-90% of the link capacity, thus heavily congesting the link. For simulations with TCP cross-traffic, each collection of nodes $CT_1$, $CT_2$, $CT_3$, and $CT_4$ is attached to four FTP sources which generates TCP cross-traffic over the congested links. The utilization of each link is in the range of 80-90%. The TCP cross-traffic consists of packets with different sizes such as 600, 800, 1000 and 1200 bytes. The probe-traffic sender PT, sends packet-trains of length $n = 48$ with 1,500 bytes packet-size at an average rate of 50 Kb/s. These packet-trains are enveloped by small packets of size 45 bytes. The inter-packet spacing in alternate packet-trains is initialized to two different values $a$ and $b$ which gives average initial spacing of $\bar{x}_a^{(0)}$ and $\bar{x}_b^{(0)}$. These values are chosen such that Lemma 1 condition is satisfied.

Every simulation has 4 phases, one for each of the congested links (hop-by-hop). In the first phase, the packet-trains are dropped at router $R_2$ (sink-node) and in the next phases, they are dropped successively at router $R_3$, $R_4$, and $R_5$. In each phase, the probe-traffic receiver PT computes the average inter-envelope-packet spacing $E_l^{(k,a)}$ and $E_l^{(k,b)}$ by alternate sampling of the envelope-packet-pairs at the receiver. These values, when applied to (4.11), (4.12), (7.7), and (7.8) provides the estimate of capacity and available bandwidth of the particular congested sink-node of the phase.

All the simulations run for approximately 100 packet-trains. The simulation results for both CBR and TCP cross-traffic are summarized in Table III and Table IV respectively. The relative estimation error $\tilde{e}_C$ in capacity estimation is less than 10% for all the cases and in most of the cases it is within 5%. In case-V where the bottleneck link precedes the tight link, capacity estimation has high errors for the

Table III. Relative Estimation Error $\tilde{e}$ with CBR Cross-Traffic.

| | Relative estimation error $\tilde{e}$ | | | |
|---|---|---|---|---|
| | Case-I | Case-II | Case-III | Case-IV |
| $C_1$ | 01.02% | 01.38% | 01.29% | 0.53% |
| $C_2$ | 00.25% | 00.92% | 00.95% | 00.06% |
| $C_3$ | 04.90% | 03.03% | 01.11% | 00.89% |
| $C_4$ | 00.62% | 02.39% | 06.18% | 03.60% |
| $A$ | 03.94% | 01.11% | 00.91% | 06.44% |

links that follow the bottleneck link, which is a common problem for all measurement methods. Envelope over estimates the base capacities of these links and under estimates the available bandwidth in these links. This is in line of our understanding, as the inter-envelope-packet spacing after the bottleneck link is so large that it is preserved through the following links.

The errors in estimation depends more on the relative congestion in the links after the bottleneck link and the expansion of packet-train length in the bottleneck link. Sometimes, after the bottleneck link we get good results (as in case I-IV) and sometimes we do not. So we can say that the results are more random once bottleneck link is reached. However that randomness would be in one direction: over estimation of base capacity and under estimation of available bandwidth. This limits Envelope in locating the bottleneck link if there are two such links with almost the same capacity. The same applies to the tight links also.

We can stop Envelope once it hits the bottleneck link. We observed that once bottleneck is reached ($E_l^{(k,a)} - E_l^{(k,b)}$) is almost constant. On an average, both samples

Table IV. Relative Estimation Error $\tilde{e}$ with TCP Cross-Traffic.

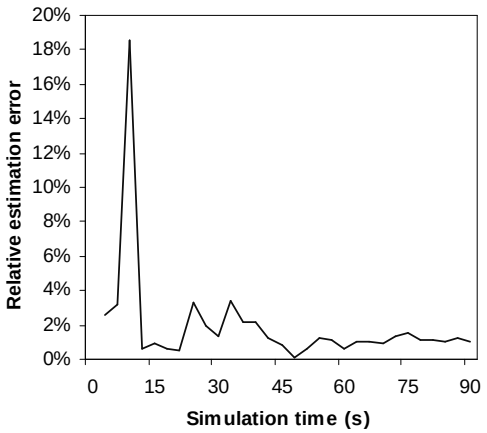
| | Relative estimation error $\tilde{e}$ | | | | |
|---|---|---|---|---|---|
| | Case-I | Case-II | Case-III | Case-IV | Case-V |
| $C_1$ | 6.93% | 2.63% | 0.75% | 4.33% | 6.52% |
| $C_2$ | 0.63% | 3.14% | 1.41% | 3.40% | 5.52% |
| $C_3$ | 3.12% | 5.78% | 0.13% | 2.23% | 33.10% |
| $C_4$ | 5.15% | 3.24% | 1.69% | 6.52% | >100% |
| $A$ | 5.73% | 10.73% | 6.82% | 8.69% | 10.57% |

are similarly expanded and hence we get utilization $\rho_k$ almost one:

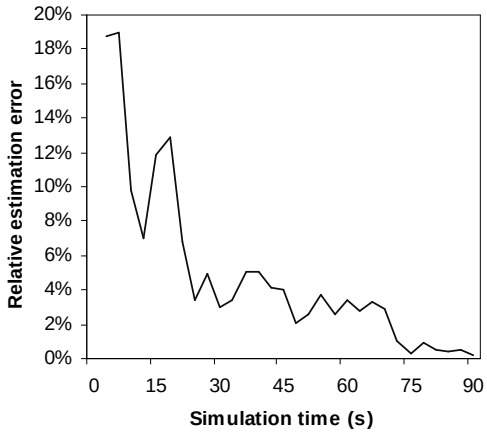
$$\rho_k = \lim_{l\to\infty} \tilde{\rho}_k^l = \lim_{l\to\infty} \frac{E_l^{(k,a)} - E_l^{(k,b)}}{E_l^{(k-1,a)} - E_l^{(k-1,b)}} \approx 1. \tag{4.15}$$

This is the reason for under estimation of available bandwidth. We can use this as a terminating condition for Envelope.

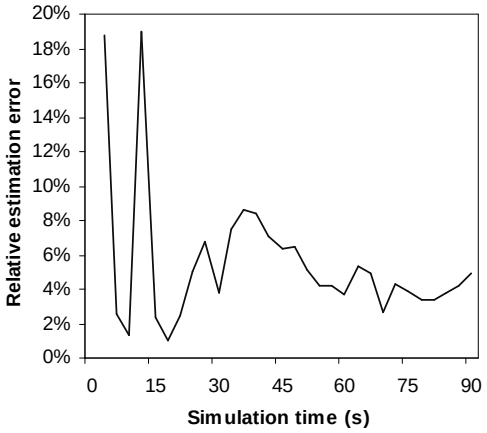
The convergence of our new method Envelope to accurate estimates in bandwidth estimation is confirmed by the graphs in figures 17-25 for different cases with CBR cross-traffic and TCP cross-traffic. The figures 17-20 present the convergence in bandwidth estimation for the four cases with CBR cross-traffic. Similarly, the figures 21-25 are for five different cases with TCP cross-traffic. From these graphs we observe that in all the cases $\tilde{e}$ is within 10% after the first 100 seconds of simulation, and in some cases it even converges below 5%. We also observe that the convergence is faster in higher capacity links in comparison to lower capacity links. This can be attributed to the fact that the relative available bandwidth is more in higher capacity links and hence, the noise in the estimation is less. In these graphs, the convergence of estimation error $\tilde{e}$ with simulation time validates the accuracy of our estimation method Envelope and also confirms its asymptotic nature.



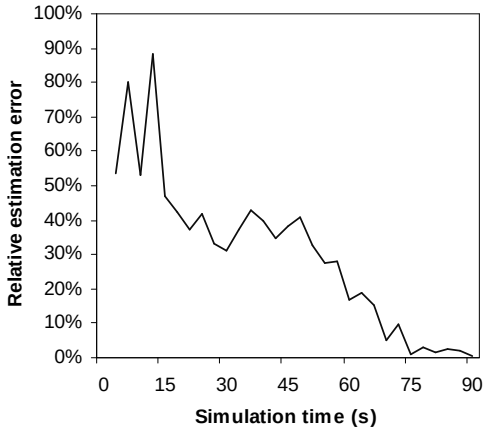

(a) $\tilde{e}_C$, $C_1 = 2$ Mb/s



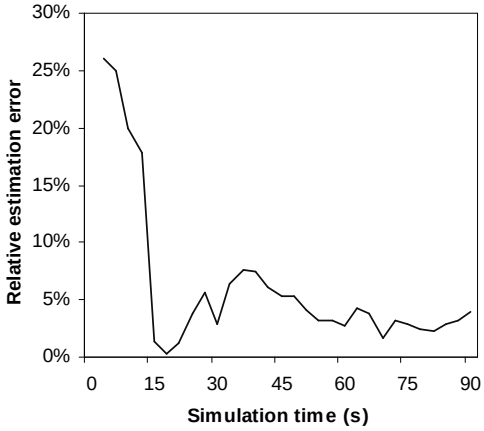

(b) $\tilde{e}_C$, $C_2 = 1.5$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 0.8$ Mb/s

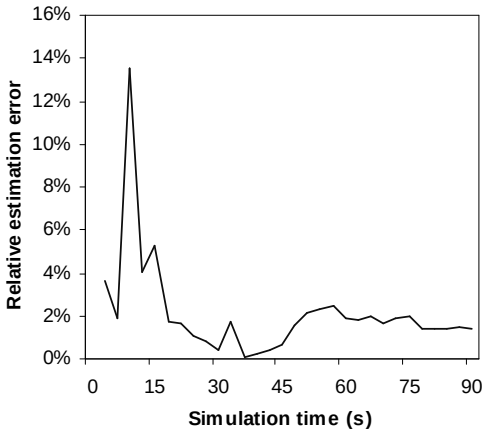



(d) $\tilde{e}_C$, $C_4 = 1.5$ Mb/s

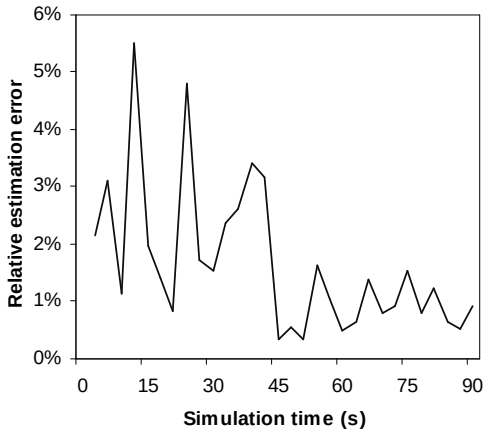



(e) $\tilde{e}_A$, $A = 160$ Kb/s in $C_3$ (c)

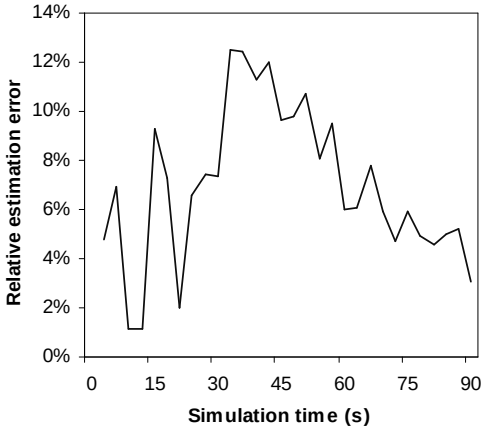
Fig. 17. CBR cross-traffic. Case-I: Bottleneck link is the tight link.



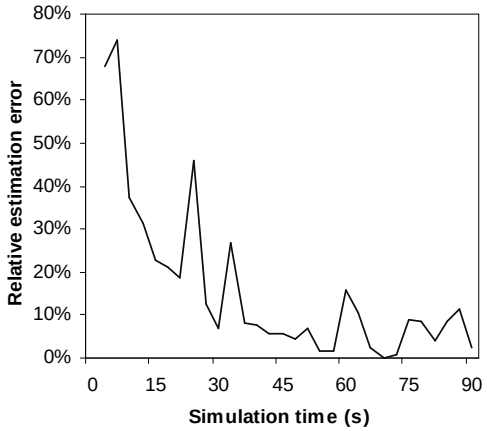

(a) $\tilde{e}_C$, $C_1 = 2$ Mb/s



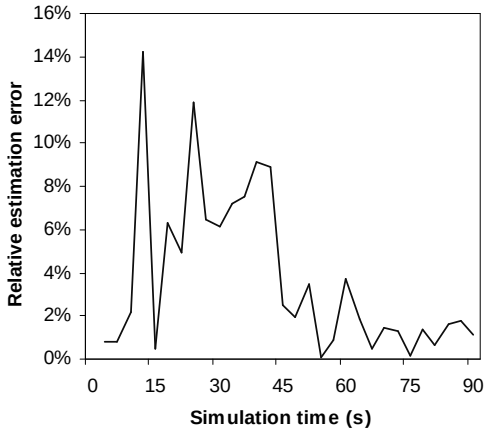

(b) $\tilde{e}_C$, $C_2 = 1.5$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 0.8$ Mb/s

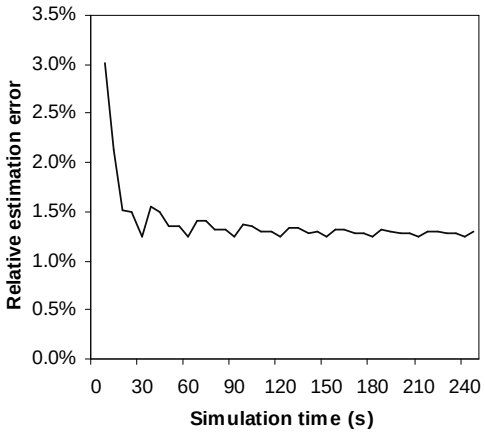



(d) $\tilde{e}_C$, $C_4 = 1.5$ Mb/s

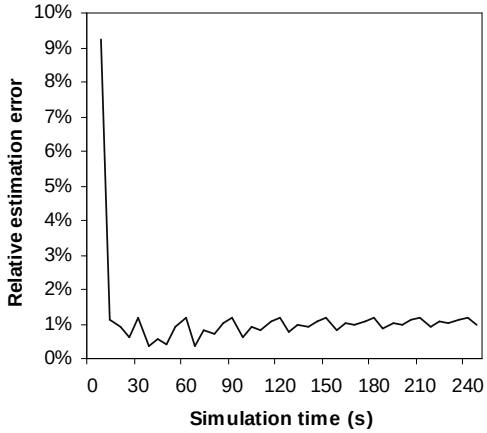



(e) $\tilde{e}_A$, $A = 250$ Kb/s in $C_2$ (f)

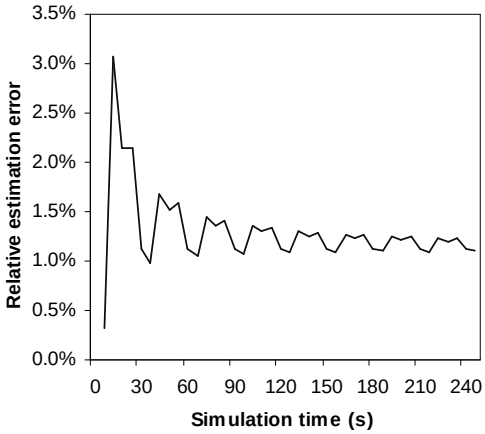
Fig. 18. CBR cross-traffic. Case-II: Bottleneck link is after the tight link.



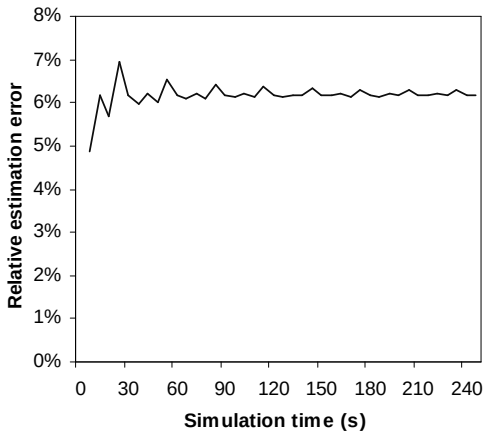

(a) $\tilde{e}_C$, $C_1 = 20$ Mb/s



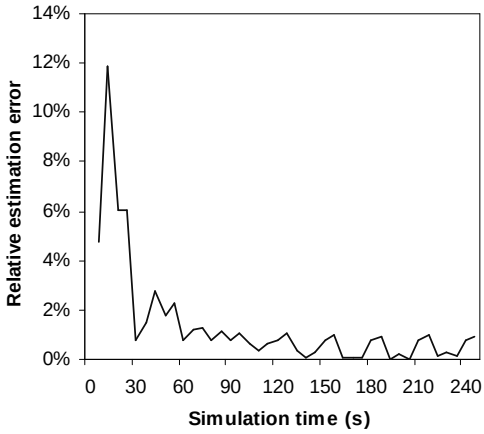

(b) $\tilde{e}_C$, $C_2 = 15$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 8$ Mb/s

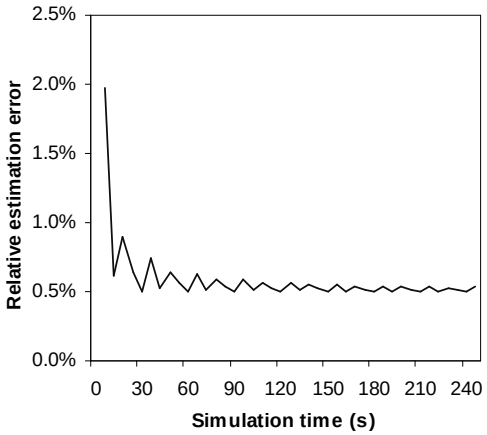



(d) $\tilde{e}_C$, $C_4 = 15$ Mb/s

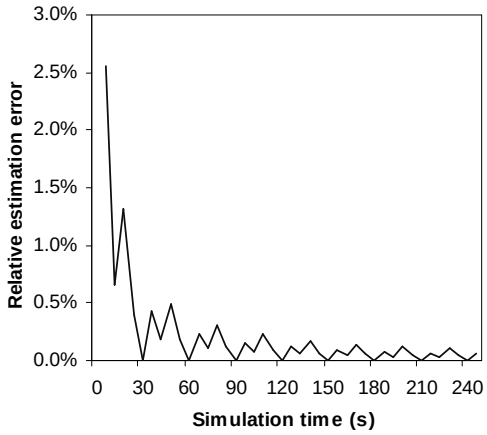



(e) $\tilde{e}_A$, $A = 1.6$ Mb/s in $C_3$ (k)

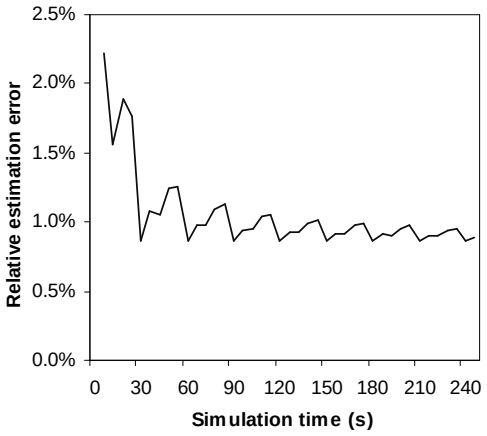
Fig. 19. CBR cross-traffic. Case-III: Bottleneck link is the tight link.



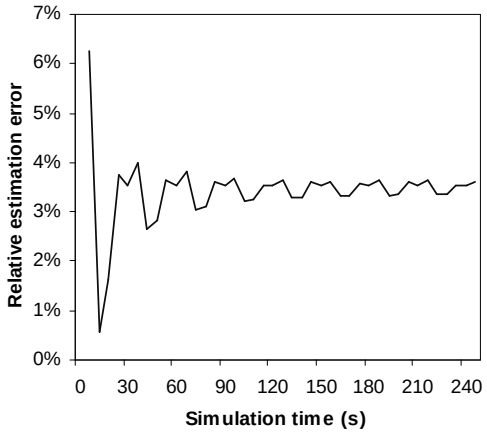

(a) $\tilde{e}_C$, $C_1 = 20$ Mb/s



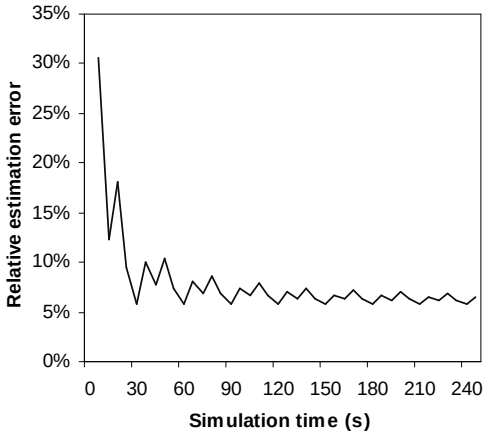

(b) $\tilde{e}_C$, $C_2 = 15$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 8$ Mb/s

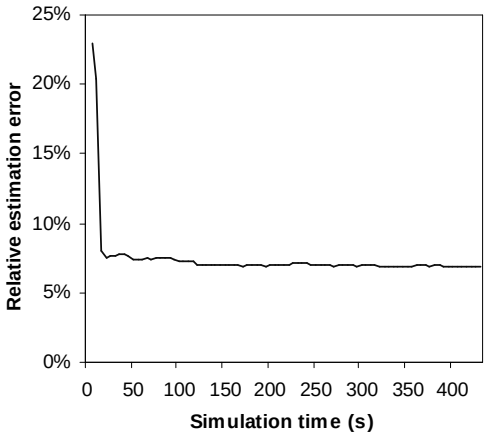



(d) $\tilde{e}_C$, $C_4 = 15$ Mb/s

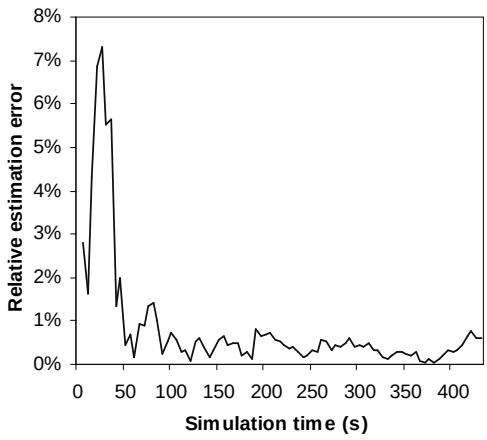



(e) $\tilde{e}_A$, $A = 2.5$ Mb/s in $C_2$ (n)

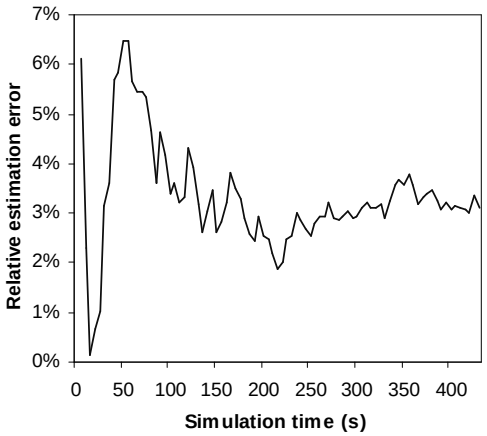
Fig. 20. CBR cross-traffic. Case-IV: Bottleneck link is after the tight link.



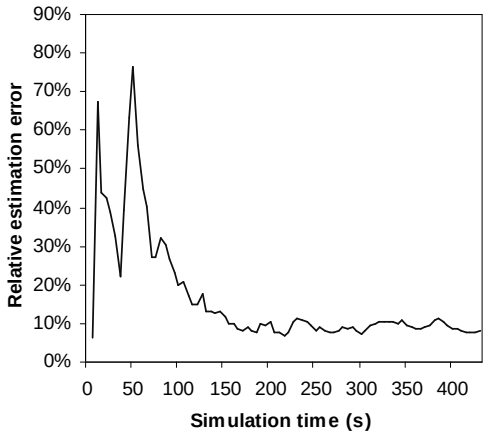

(a) $\tilde{e}_C$, $C_1 = 2$ Mb/s



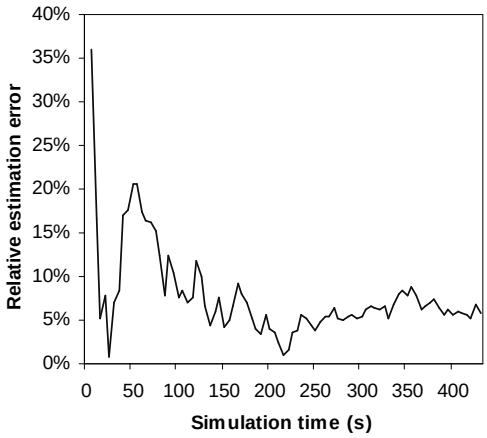

(b) $\tilde{e}_C$, $C_2 = 1.5$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 0.8$ Mb/s

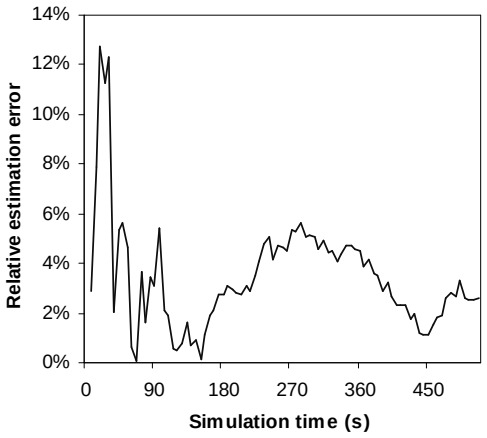



(d) $\tilde{e}_C$, $C_4 = 1.5$ Mb/s

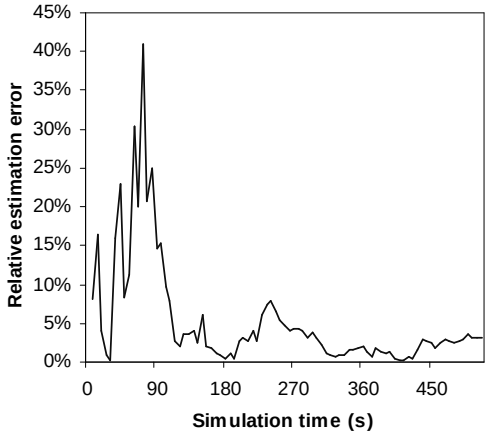



(e) $\tilde{e}_A$, $A = 160$ Kb/s in $C_3$ (c)

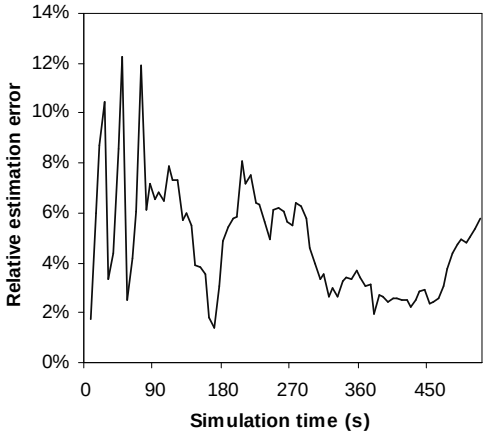
Fig. 21. TCP cross-traffic. Case-I: Bottleneck link is the tight link.



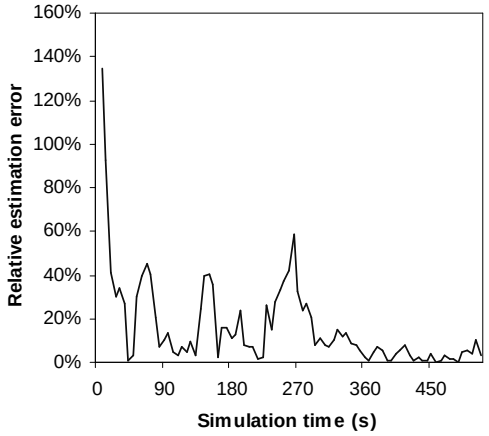

(a) $\tilde{e}_C$, $C_1 = 2$ Mb/s



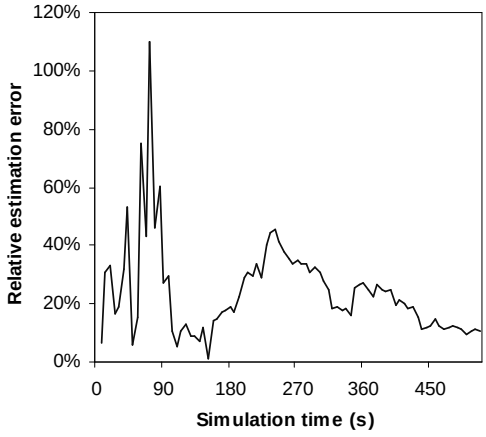

(b) $\tilde{e}_C$, $C_2 = 1.5$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 0.8$ Mb/s

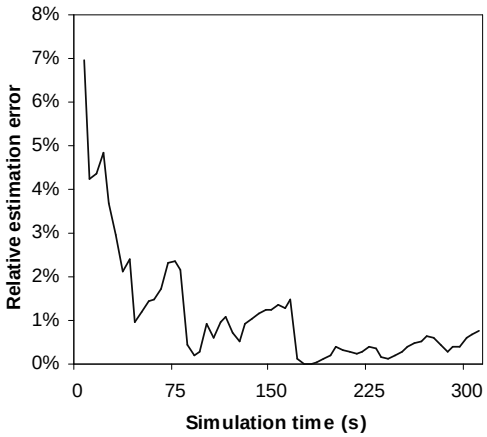



(d) $\tilde{e}_C$, $C_4 = 1.5$ Mb/s

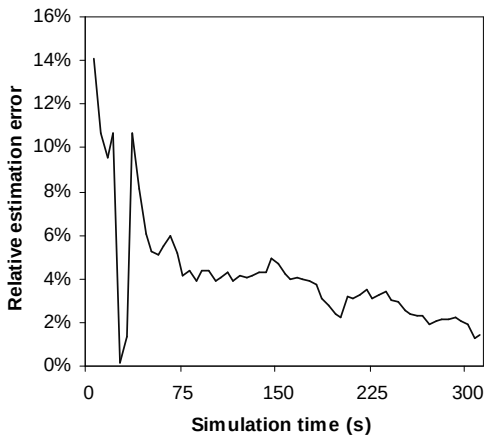



(e) $\tilde{e}_A$, $A = 250$ Kb/s in $C_2$ (f)

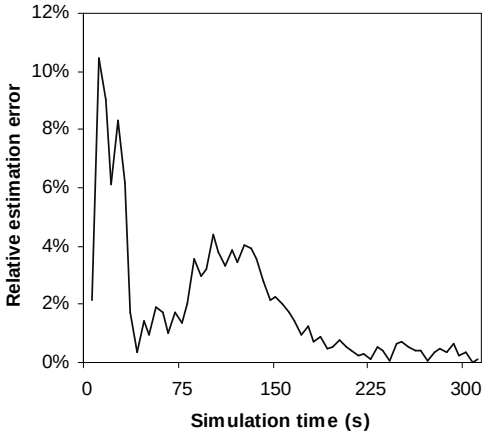
Fig. 22. TCP cross-traffic. Case-II: Bottleneck link is after the tight link.



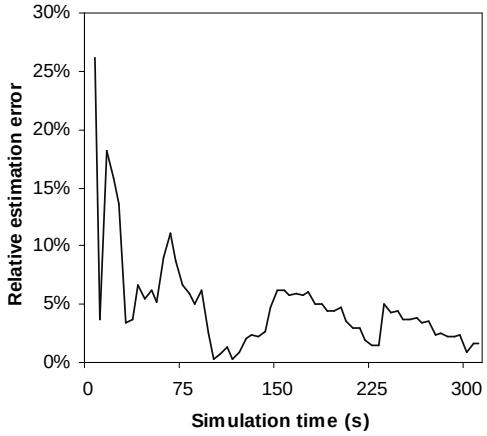

(a) $\tilde{e}_C$, $C_1 = 20$ Mb/s



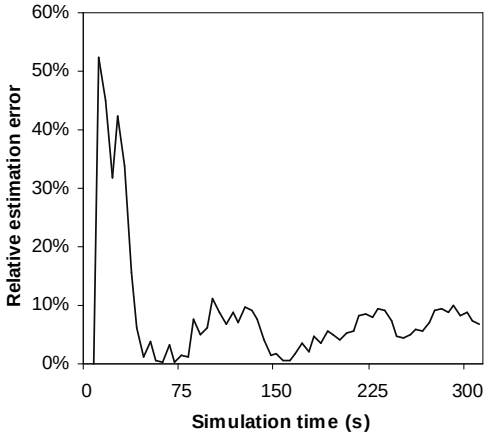

(b) $\tilde{e}_C$, $C_2 = 15$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 8$ Mb/s

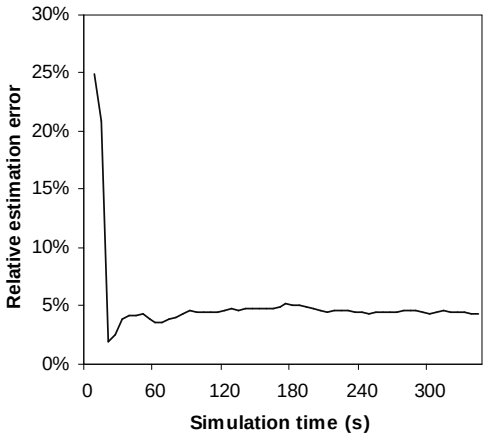



(d) $\tilde{e}_C$, $C_4 = 15$ Mb/s

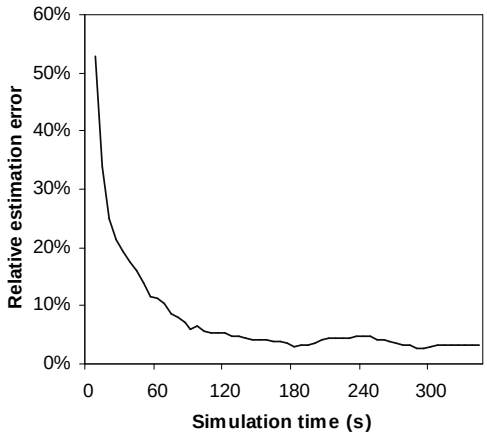



(e) $\tilde{e}_A$, $A = 1.6$ Mb/s in $C_3$ (k)

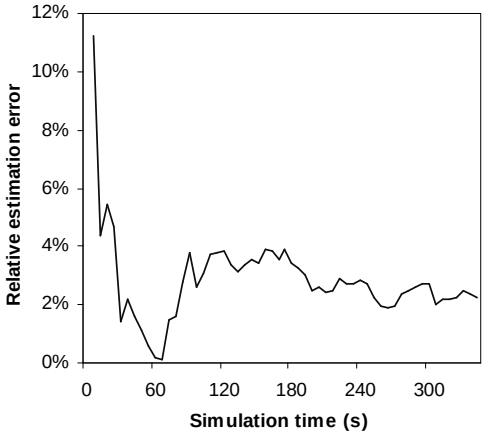
Fig. 23. TCP cross-traffic. Case-III: Bottleneck link is the tight link.



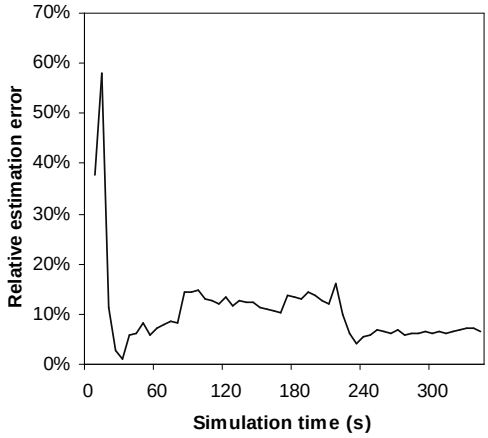

(a) $\tilde{e}_C$, $C_1 = 20$ Mb/s



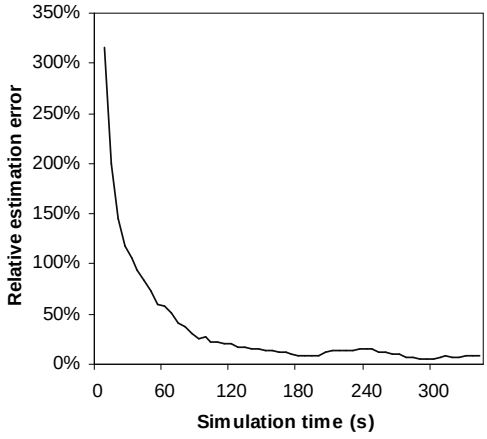

(b) $\tilde{e}_C$, $C_2 = 15$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 8$ Mb/s

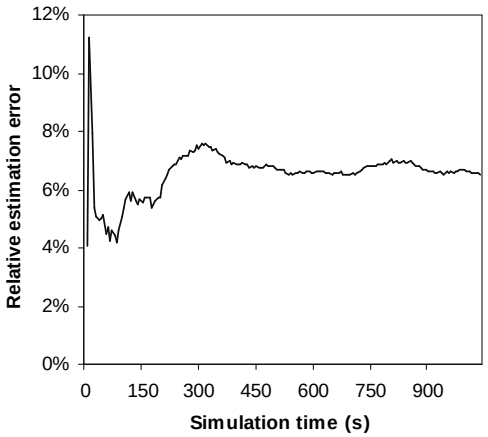



(d) $\tilde{e}_C$, $C_4 = 15$ Mb/s

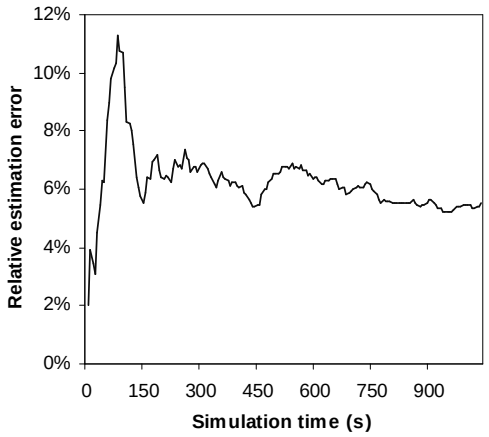



(e) $\tilde{e}_A$, $A = 2.5$ Mb/s in $C_2$ (n)

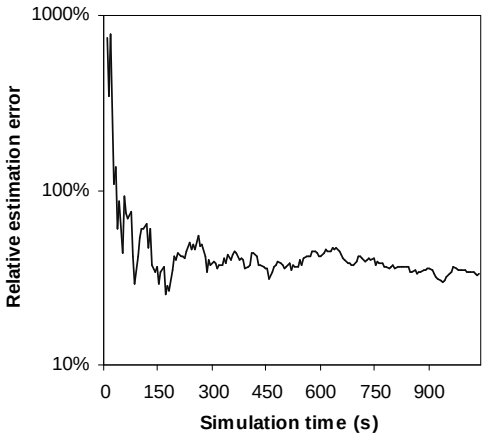
Fig. 24. TCP cross-traffic. Case-IV: Bottleneck link is after the tight link.



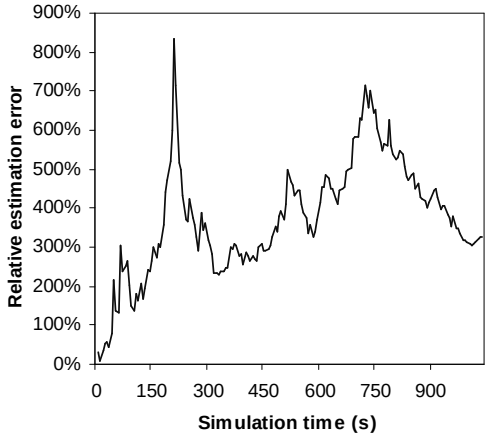

(a) $\tilde{e}_C$, $C_1 = 2$ Mb/s



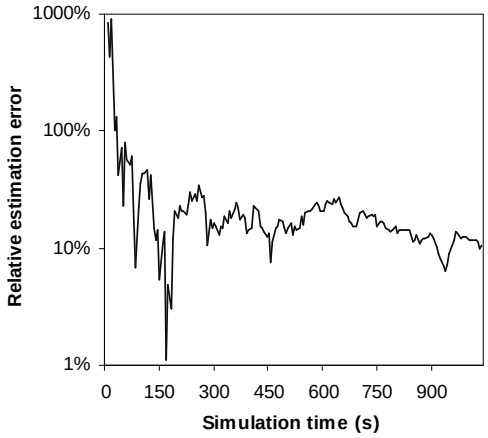

(b) $\tilde{e}_C$, $C_2 = 0.8$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 1.5$ Mb/s




(d) $\tilde{e}_C$, $C_4 = 2$ Mb/s




(e) $\tilde{e}_A$, $A = 250$ Kb/s in $C_3$ (c)

Fig. 25. TCP cross-traffic. Case-V: Bottleneck link is before the tight link.

# CHAPTER V

# COMPARISON

In this chapter, we compare our method Envelope with the other existing bottleneck and available bandwidth estimation methods with respect to their estimation accuracy. We perform simulations in NS-2 using the four-congested node topology shown in Fig. 16. We simulate the first four cases suggested in Table II for all the methods and compare their estimation results with those of our method Envelope. We also analyze the estimation techniques in some of these methods to arrive at some interesting observations.

## A. Available Bandwidth Estimation Methods

Here, we compare the results of available bandwidth estimation methods such as IGI [13], Spruce [14], and Pathload [15], with that of our method Envelope. We have used the NS-2 implementation of these methods for simulations. The comparison of the relative estimation error $\tilde{e}_A$ of these methods for different cases is shown in Table V and Table VI. Table V shows the results obtained with CBR cross-traffic while Table VI shows the results obtained with TCP cross-traffic.

In all the simulations, the nodes are heavily congested with the utilization of about 80-90%. The results confirm that Envelope outperforms both IGI and Spruce by huge margins in all the cases where utilization of the congested links is high. Pathload results are comparable with that of Envelope only when the link capacities are high as in case-III and case-IV. Both Spruce and IGI use the known bottleneck bandwidth $C_3$ for their estimations of available bandwidth $\tilde{A}$, whereas Envelope computes the capacity of the congested links and uses it in the estimation of $\tilde{A}$. Pathload estimates a range of available bandwidth without any prior knowledge of the capacities and

Table V. Comparison of Available Bandwidth Estimation Methods: CBR Cross-Traffic.

| | Relative estimation error $\tilde{e}_A$ | | | |
|---|---|---|---|---|
| | Envelope | Pathload | Spruce | IGI |
| Case-I | 03.94% | 31.25% | >100% | 86.88% |
| Case-II | 01.11% | 25.00% | 68.47% | >100% |
| Case-III | 00.91% | 10.31% | >100% | >100% |
| Case-IV | 06.44% | 08.33% | 67.11% | 88.89% |

Table VI. Comparison of Available Bandwidth Estimation Methods: TCP Cross-Traffic.

| | Relative estimation error $\tilde{e}_A$ | | | |
|---|---|---|---|---|
| | Envelope | Pathload | Spruce | IGI |
| Case-I | 08.23% | 49.90% | >100% | >100% |
| Case-II | 10.73% | 33.33% | >100% | >100% |
| Case-III | 06.82% | 09.00% | >100% | >100% |
| Case-IV | 08.69% | 10.00% | 86.57% | 90.00% |

we have taken the mean of this range for our analysis. Even with the accurate knowledge of bottleneck bandwidth, the results of Spruce and IGI are not good for heavily congested nodes. So we further analyze these methods to find out the reasons for their failure in such cases.

### 1. Spruce

Spruce is based on the probe gap model (PGM) [14] which is derived for a single bottleneck link that is both, the narrow and the tight link along the path. Spruce

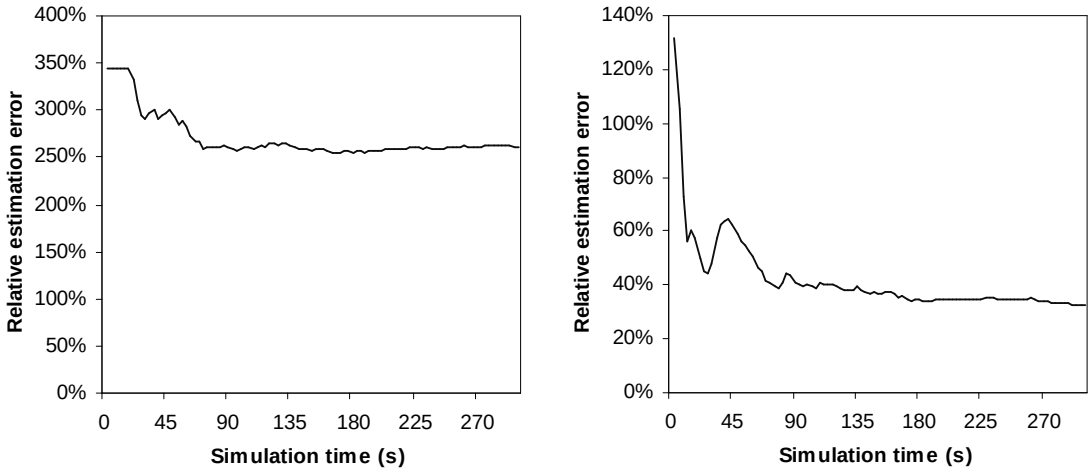
uses a modified packet-pair technique, which ensures that the packets of every packet-pair queue behind each other at the bottleneck node. It measures the intra-packet-pair spacing $y_n$ at the receiver to estimate the available bandwidth $A_n$ using the equation [14]:

$$A_n = C\left(1 - \frac{y_n - \Delta}{\Delta}\right). \tag{5.1}$$

We observe that the analysis neglects the interference of cross-traffic with the probe-gap at the nodes other then the bottleneck node over the entire path. Hence, with heavily congested (80-90%) pre- and post- bottleneck nodes, the estimation accuracy is bound to be low. This is illustrated by the graphs in Fig. 26 which plots $\tilde{e}_A$ for different link utilizations. Fig. 26(a) is for the case where the nodes are heavily utilized, whereas Fig. 26(b) is for the lightly utilized nodes. We can observe that the estimation error falls from 250% to almost 35% when utilization is reduced from 80% to 40%. This confirms the limitation of Spruce to be used in the heavily utilized paths. This explains the high estimation errors in Case-I and Case-III as shown in Table V and Table VI. Moreover, if the bottleneck link is not the tight link or the bottleneck bandwidth is not known with good accuracy, then Spruce does not converge to good estimates. This is evident from the results for Case-II and Case-IV. Similar observations were also made in [9].

### 2. IGI

Similarly, for cases where the bottleneck link is not the tight link (Case-II and Case-IV), the estimation accuracy of IGI also decreases due to its dependence on the knowledge of bottleneck capacity. IGI [13] is based on a gap model whose analysis with respect to a single congested node neglects the effect of pre- and post- bottleneck cross traffic. According to [13], the IGI algorithm sends a sequence of packet-trains




(a) $\tilde{e}_A$, utilization = 80% (b) $\tilde{e}_A$, utilization = 40%

Fig. 26. Estimation accuracy of Spruce in heavy and lightly loaded bottleneck link.

with increasing inter-packet spacing in each packet-train until the average inter-packet spacing at the receiver is equal to the average initial inter-packet spacing at the sender. At this point, the noise introduced by bottleneck cross-traffic is zero mean and the probing rate is the available bandwidth. Here, the analysis completely neglects the pre- and post- bottleneck cross-traffic noise that might increase or decrease the average probe-gap at the receiver. This is the reason, IGI fails to estimate $A$ accurately in Case-I and Case-III, where all the nodes are heavily congested. To emphasize this point, we simulate Case-I (TCP) with reduced cross-traffic at routers $R_1$, $R_2$ and $R_4$ (40% utilized). We find that the relative estimation error $\tilde{e}_A$ is 56.29%, which is almost 50% less than the estimation error (105.36%) with heavily loaded routers. This proves that IGI is not a useful method in heavily utilized paths.

### 3. Pathload

As mentioned earlier, Pathload is based on the observation that if the probe-rate is higher than the available bandwidth, then the inter-packet spacing of probe-traffic

Table VII. Comparison of Bottleneck Bandwidth Estimation Methods.

| Bottleneck Capacity (Mb/s) | Relative estimation error $\tilde{e}_C$ | | | |
|---|---|---|---|---|
| | CBR | | TCP | |
| | Envelope | CapProbe | Envelope | CapProbe |
| Case-I: 0.8 | 04.90% | 71.22% | 03.86% | 87.27% |
| Case-III: 8 | 01.11% | 71.43% | 00.13% | 85.19% |

at the receiver shows an increasing trend. However, with heavy cross-traffic in the entire path, this trend might change before it reaches the receiver. This is evident from the results in Table V and Table VI, which shows that the relative estimation error in the cases (Case-I and Case-II) with small capacity links is higher than those in the cases (Case-III and Case-IV) with large capacity links. This can be attributed to the fact that the relative available bandwidths in all the links are higher in the latter cases in comparison to the former ones. Hence, in the Case-III and Case-IV, the trend of inter-packet spacing of probe-traffic at the tight link is better preserved until the receiver is reached and we get better estimates. We can also observe that Pathload performs almost similar to Envelope in two cases III and IV, however, in other two cases Envelope performs better.

## B. Bottleneck Bandwidth Estimation Methods

In this section, we compare the results of CapProbe [16], a recent bottleneck bandwidth estimation method, with that of our method Envelope. The results are obtained through NS-2 simulations. The comparison of the relative estimation error $\tilde{e}_C$ of the two methods, in Case-I and Case-III with CBR and TCP cross-traffic, is shown in Table VII. These results confirm that Envelope performs better than CapProbe in the

Table VIII. Relative Estimation Error $\tilde{e}_C$ in CapProbe with different Utilization $\rho$ of Congested Links.

| $\rho$ | 50.00% | 60.00% | 70.00% | 80.00% | 90.00% |
|---|---|---|---|---|---|
| $\tilde{e}_C$ | 00.00% | 11.98% | 11.98% | 87.27% | 81.66% |

heavily utilized paths. Moreover, in some cases, the CapProbe algorithm suggested in [16] does not converge with runs of 100 samples and we have to use runs with more than 1000 samples to get the convergence. To better understand CapProbe's estimation technique, we further analyze it.

### 1. CapProbe

As mentioned in chapter II, CapProbe [16] is based on the observation, that the difference in the queuing delays of two back-to-back packets at the receiver reflects the transmission delay of the packets in the bottleneck link if they are never queued at any router in the path. Such queuing delays would also be the minimum among all the packet-pair samples. CapProbe uses runs of 100 samples and a minimum delay condition to filter out the queued packets. However, we found that with heavily congested paths, as in Case-I and Case-III, CapProbe does not converge with 100 sample runs. Rather, it takes more than 1000 samples to get the convergence with a relative error of about 85%. The reason for such a late convergence is the presence of four heavily congested routers in the path. Due to the heavy cross-traffic at these routers, the probability of two back-to-back packets not being queued in any of these routers in first 100 samples is much less. Moreover, we observe that the convergence is extremely random in such cases and it is very difficult to decide the number of samples to be used in the estimation algorithm. We further analyzed CapProbe with different loads on the routers of Case-I, as shown in Table VIII. We observed

that, as the utilization of the congested routers increases from 50% to 90%, the error in estimation $\tilde{e}_C$ also increases from 0% to 82%. This confirms that CapProbe performs very well in the less congested paths. However, in the heavily congested paths, CapProbe's capacity estimation is poor. Hence, there is evidence to suggest that Envelope outperforms CapProbe in the heavily congested paths.

# CHAPTER VI

## EVALUATION OF METHODOLOGY

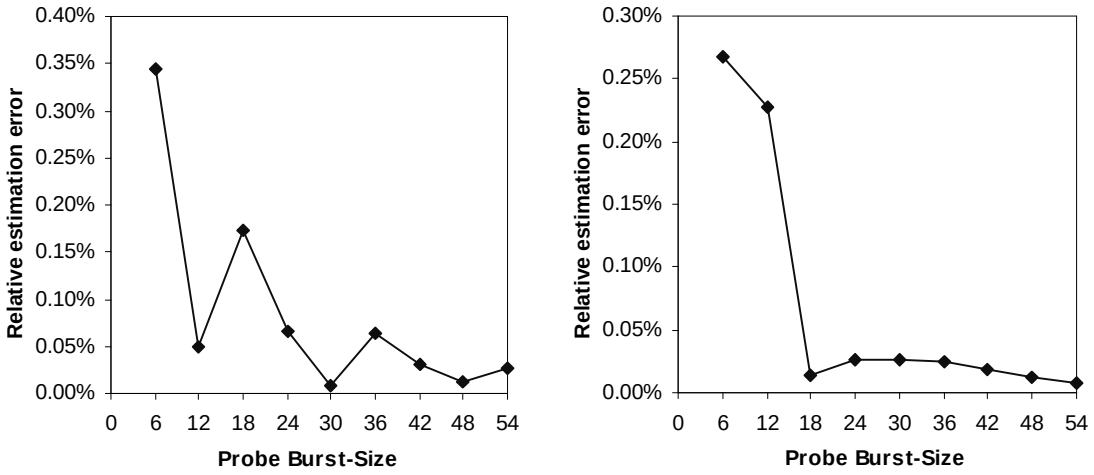
In this chapter, we evaluate various probe-traffic parameters which affect the accuracy of Envelope. Probe-traffic parameters, such as probe-traffic burst-size $n$ (packet-train length), probe-traffic packet-size $q_p$, and the initial inter-packet spacing $x^{(0)}$, can affect the spacing preservation strategy of Envelope and hence, the estimation accuracy. Therefore, we study these parameters and their effect on the estimation accuracy through simulations in NS-2, and obtain a range of values for these parameters that provide good estimation results. For simulations, we use the multi-congested topology shown in Fig. 16 for Case-I from Table II.

### A. Burst Size

The main idea behind the method Envelope is to preserve the inter-packet spacing $y^{(k)}$ of probe packets in the path-suffix of a congested router $R_k$ using small envelope-packets. As suggested in Lemma 2, this is attained by having enough number of packets $n$ in a probe-burst between the envelope-packets. As a result, the noise introduced by the cross-traffic in the path-suffix is zero-mean and the spacing is preserved. Hence, in this subsection, we study the effect of the probe-traffic burst-size $n$ on the accuracy of the preservation technique in Envelope. In order to measure the error in the space-preservation, we define relative estimation error in preservation $\tilde{e}_p$ as:

$$\tilde{e}_p = \lim_{l \to \infty} \frac{|E_l^{(k)} - \tilde{E}_l^{(k)}|}{E_l^{(k)}}, \tag{6.1}$$

where $E_l^{(k)}$ is defined in (4.9) as an average inter-packet spacing of $l$ probe-bursts. Here, $E_l^{(k)}$ is the actual spacing sampled after the bottleneck router $R_k$, while $\tilde{E}_l^{(k)}$



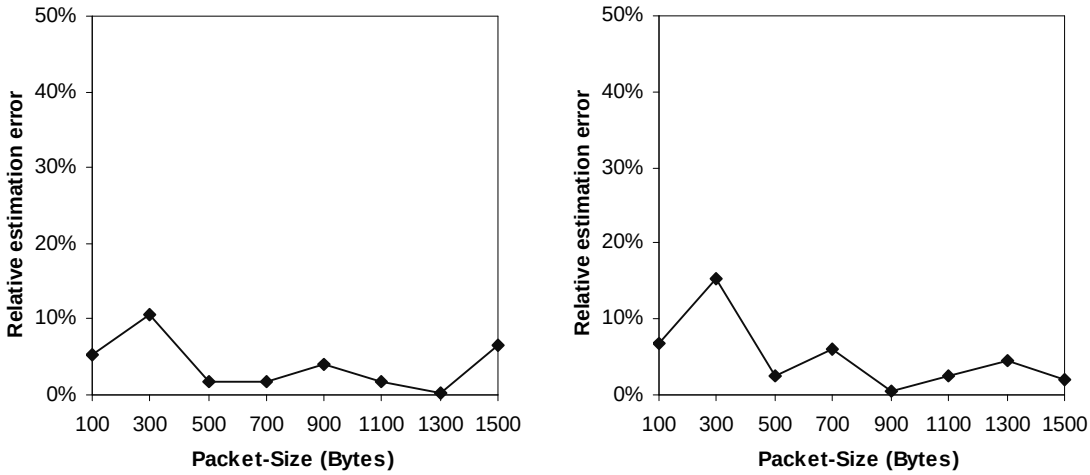

(a) $\tilde{e}_p^a$, where $x_a^{(0)} = 5.0$ ms (b) $\tilde{e}_p^b$, where $x_b^{(0)} = 5.5$ ms

Fig. 27. Relative estimation error $\tilde{e}_p$ variation with increasing probe burst-size $n$.

is estimated spacing sampled at the receiver. The convergence of relative estimation error $\tilde{e}_p$ for different probe burst-sizes $n$, and for both the signals $x_a^{(0)}$ and $x_b^{(0)}$ is shown in Fig. 27. Here, we sample the envelope-packet-pairs after the bottleneck router $R_3$ to obtain $E_l^{(3)}$, and at the receiver PT to obtain $\tilde{E}_l^{(3)}$ for both the signals alternately. Then, (6.1) is used to obtain the convergence of $\tilde{e}_p^a$ and $\tilde{e}_p^b$ with $n$. We observe that, as the burst-size increases, the relative error in space-preservation $\tilde{e}_p$ decreases and for bursts-size greater than 36, it converges below 0.05%. We further found that a small error of about 0.1-0.2% can have a large impact on the estimation accuracy of Envelope. Hence, we suggest a burst-size of $n \geq 36$ can be considered as a optimal range for good estimation accuracy. In all the simulations in the previous sections, we used a burst-size of $n = 48$.

## B. Probing Packet Size

Here, we study the impact of probe-traffic packet-size $q_p$ on the estimation accuracy of Envelope. Through NS-2 simulations, we compare the relative estimation error $\tilde{e}$



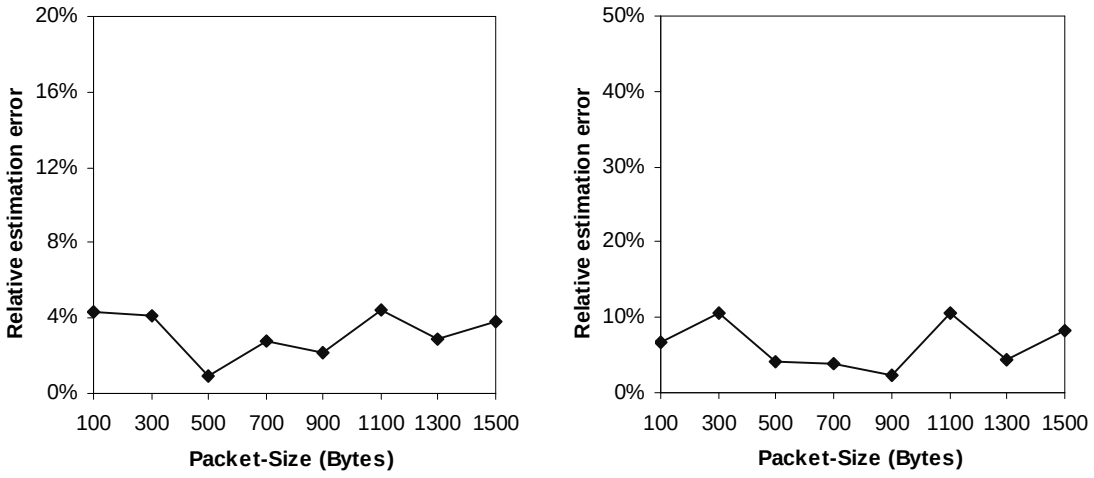

(a) $\tilde{e}_C$ (b) $\tilde{e}_A$

Fig. 28. Relative error $\tilde{e}$ in bottleneck and available bandwidth estimation with increasing probe-traffic packet-size $q_p$ and CBR cross-traffic.

in the bottleneck and available bandwidth estimation for different $q_p$ varying from 100 to 1500 bytes (maximum packet-size in NS-2). The simulation results, $\tilde{e}_C$ and $\tilde{e}_A$ with increasing $q_p$, for Case-I with CBR and TCP cross-traffic are shown in Fig. 28 and Fig. 29 respectively. From the graphs, we observe that errors $\tilde{e}_C$ and $\tilde{e}_A$ are always below 10% for almost all packet-sizes. In order to determine the dependency of Envelope on the probe packet-size, we calculated the standard deviation $\sigma$ of the set of N = 8 samples of $\tilde{e}$ for different packet-sizes as:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\bar{e} - \tilde{e}_i)^2}, \tag{6.2}$$

where $\bar{e}$ is the mean of the N samples of $\tilde{e}$. The standard deviation $\sigma$ for Case-I with CBR and TCP cross-traffic are shown in Table IX. The $\sigma$ in $\tilde{e}_C$ with TCP cross-traffic is $\sigma_C = 0.0123$ or 1.23% , while for $\tilde{e}_A$ is $\sigma_A = 0.0315$ or 3.15%. Similarly, for CBR cross-traffic also, the $\sigma_C$ and $\sigma_A$ are below 5%. Such low standard deviation $\sigma$ values suggests that the Envelope's estimation accuracy is almost independent of the probe




(a) $\tilde{e}_C$ (b) $\tilde{e}_A$

Fig. 29. Relative error $\tilde{e}$ in bottleneck and available bandwidth estimation with increasing probe-traffic packet-size $q_p$ and TCP cross-traffic.

Table IX. Standard Deviation $\sigma$ of Relative Estimation Error $\tilde{e}$.

| Case-I | Standard deviation $\sigma$ in $\tilde{e}$ | | | |
|---|---|---|---|---|
| Capacity | CBR | | TCP | |
| (Mb/s) | $\sigma_C$ | $\sigma_A$ | $\sigma_C$ | $\sigma_A$ |
| $C = 0.8; A = 0.16$ | 03.42% | 04.64% | 01.23% | 03.15% |

packet-size.

The transmission delay $\Delta_k$ of the probe packet in a link with capacity $C_k$ depends on the packet-size $q_p$ ($\Delta_k = q_p/C_k$): Larger the packet-size $q_p$, higher the transmission delay $\Delta_k$. Thus, in order to satisfy the condition mentioned in Lemma 1, we need to vary the initial inter-packet spacing $x_a^{(0)}$ and $x_b^{(0)}$ of probe-traffic appropriately: smaller the packet-size $q_p$, smaller the spacing $x_a^{(0)}$ and $x_b^{(0)}$. Therefore, with a small packet-size the probability of capturing the noise introduced by cross-traffic with a fixed number of samples decreases. Thus, we need to run simulations for larger

number of samples to obtain good estimation accuracy. This is more computation intensive. Hence, we suggest using large size packets in the probe-traffic. We used packets of size $q_p = 1500$ bytes for all the simulation in the previous sections.

## C. Inter-Packet Spacing

In this subsection, we analyze one of the important parameters of the probe-traffic, the initial inter-packet spacing $x^{(0)}$. There are various factors which bound $x^{(0)}$ such as Lemma 1, probe packet-size $q_p$, and probe-rate $r_p$. As explained previously, the probe-traffic in Envelope consists of two alternate packet-trains with different inter-packet spacings $x_a^{(0)}$ and $x_b^{(0)}$. The burst-time $T_B$ of the packet-trains is decided by the burst-size $n$ of the train, while the idle-time $T_I$ between the two trains is decided by the probe-rate $r_p$. These relations can be summarized in the following equations:

$$T_p = T_B + T_I = \frac{nq_p}{r_p}. \tag{6.3}$$

$$T_B = (n-1)\, x^{(0)}. \tag{6.4}$$

$$T_I = \frac{nq_p}{r_p} - (n-1)\, x^{(0)}. \tag{6.5}$$

The probe-rate $r_p$ has to be small so that the probe-traffic does not interfere with the other flows in the network. Therefore, the idle-time $T_I$ has to be high, assuming that the burst-size $n$ and the packet-size $q_p$ are fixed according to the previous sections. As suggested in [9], in order to have PASTA [28] sampling at the receiver, $T_I$ should be derived from a exponential random variable. This satisfies the assumption (2.6) that the arbitrary cross-traffic has finite asymptotic time average. However, during the simulations, we find that even if we relax the assumption of PASTA sampling and send probe-traffic at a low rate, we obtain good estimates.

According to Lemma 1, the spacing $x^{(0)}$ is set to a small value of the order of 100

$\mu$s. We further observed that even if the condition mentioned in Lemma 1 is violated in some congested routers in the path, the estimation results are good with $\tilde{e}$ around 5-10%. From extensive simulations, we found that the two signals $x_a^{(0)}$ and $x_b^{(0)}$ can be within the range of 25-50% of each other. For example, if $x_a^{(0)} = 400$ $\mu$s, then $x_b^{(0)}$ can be taken from the range 100-200 $\mu$s. If we use these relations in the simulations while deciding the probe-traffic parameters, the estimation results are good, as shown in the previous sections.

# CHAPTER VII

## MODIFIED ENVELOPE METHOD

In this chapter, we discuss a slight variant of the method Envelope that simplifies the estimation equations and provide similar accuracy as the Envelope.

### A. New Methodology

Recall that in Envelope, two small envelope-packets surround the probing packet-train and preserve the spacing, between the first and the last probing packet of the train, at each router in the path-suffix. There is a minor problem in the orientation of the small envelope packets with respect to the packet-train. For estimating the mean inter-departure spacing $E[\bar{y}_i^{(k)}]$ at router $R_k$ from the spacing $z_i^(k)$ sampled at the receiver using (4.7), we assume that the leading envelope-packet do not get dispersed away from the first probe-train packet in the routers before $R_k$ in the path-prefix. However, we observed during simulations that this assumption does not hold true sometimes. As a result, the estimated mean inter-departure spacing $E[\bar{y}_i^{(k)}$ at router $R_k$ is slightly higher than the actual value. We further observed that such an error gets almost canceled in the estimation analysis, which involves relative difference between the estimated spacing for two different signals. Here, the assumption is that the error due to the dispersion of leading envelope-packet is same in both the signals. However, to avoid this error and to make Envelope more stable and robust, we further enhance it.

In the enhanced method, instead of sending the leading envelope-packet before the first probe-train packet, we send it after the first probe-train packet. Thus, being a small packet trailing a large packet, the leading envelope-packet always trails behind the first probe-train packet and do not get dispersed away. Hence, the mean

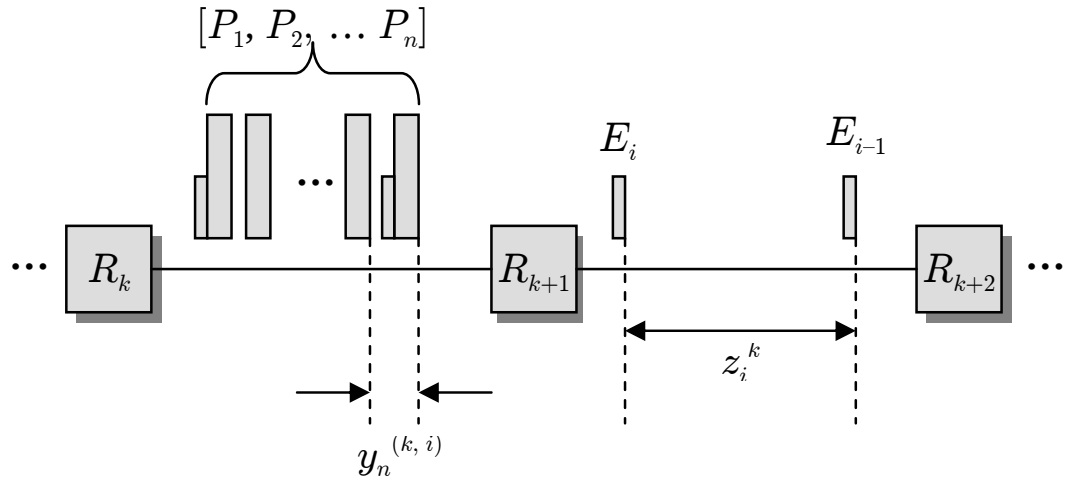



Fig. 30. A probe-burst $[P_1, P_2, ..., P_n]$ of $n$ packets enveloped by two small packets $[E_i, E_{i-1}]$ at router $R_k$.

inter-departure spacing $E[\bar{y}_i^{(k)}$ at router $R_k$ is preserved with more accuracy. The orientation of the envelope-packets with respect to the probe-train is shown in Fig. 30. With this modified approach, the estimation equations are simplified. The spacing $z_i^{(k)}$ reflects the collective inter-packet spacings of one probe-traffic burst:

$$z_i^{(k)} = (n-1)\bar{y}_i^{(k)}, \tag{7.1}$$

where $\bar{y}_i^{(k)}$ is the estimated average inter-departure spacing of $i^{th}$ probe packets at the router $k$. Now combining the recursive model (4.4) and (7.1), we obtain:

$$\bar{y}_i^{(k)} = \frac{z_i^{(k)}}{n-1} = \Delta_k + \frac{r_k(t_i)\bar{y}_i^{(k-1)}}{C_k}. \tag{7.2}$$

Define $E_l^{(k)}$ to be the average of $l$ samples of $\bar{y}_i^{(k)}$:

$$E_l^{(k)} = \frac{1}{l}\sum_{i=1}^{l} \bar{y}_i^{(k)} = \frac{1}{l}\sum_{i=1}^{l} \frac{z_i^{(k)}}{n-1}. \tag{7.3}$$

Finally, taking limits in (7.2) and (7.3), we get:

$$\lim_{l\to\infty} E_l^{(k)} = \Delta_k + \frac{E[\bar{y}_i^{(k-1)}]\bar{r}_k}{C_k}. \tag{7.4}$$

The equation (7.4) is a simpler version of (4.10) and we solve it the same way as before. During simulations, we use two alternate packet-trains with initial inter-packet spacing $x^{(0)} = a$ and $x^{(0)} = b$, and measure the signals $E_l^{(k,a)}$ and $E_l^{(k,b)}$ at the receiver by alternate sampling of envelope-packets. The following equations provide asymptotic convergence of $\Delta_k$ and $\rho_k$:

$$\lim_{l\to\infty} \tilde{\Delta}_k^l = \lim_{l\to\infty} \frac{E_l^{(k-1,a)} E_l^{(k,b)} - E_l^{(k-1,b)} E_l^{(k,a)}}{E_l^{(k-1,a)} - E_l^{(k-1,b)}} = \Delta_k. \tag{7.5}$$

$$\lim_{l\to\infty} \tilde{\rho}_k^l = \lim_{l\to\infty} \frac{E_l^{(k,a)} - E_l^{(k,b)}}{E_l^{(k-1,a)} - E_l^{(k-1,b)}} = \rho_k. \tag{7.6}$$

Using (7.5) and (7.6), capacity $C_k$ and available bandwidth $A_k$ of the link $k$ can be estimated as:

$$\lim_{l\to\infty} \tilde{C}_k^l = \lim_{l\to\infty} \frac{q}{\tilde{\Delta}_k^l} = C_k. \tag{7.7}$$

$$\lim_{l\to\infty} \tilde{A}_k^l = \lim_{l\to\infty} \left(1 - \tilde{\rho}_k^l\right) \tilde{C}_n = A_k. \tag{7.8}$$

B. Simulations

In this section, we verify the modified Envelope methodology through NS-2 simulations. We use the same network topology as given in Fig. 16. We report experimental results in five different network settings as showed in Table X. In all five cases, we use TCP cross-traffic. Table X also lists the capacity and available bandwidth of each link, where the shaded values are bottleneck capacities and available bandwidths. Note that the settings are similar to those used previously for Envelope.

In all five cases, we attached four FTP sources to each of the nodes labeled CT

Table X. Capacities and Available Bandwidths During Simulations.

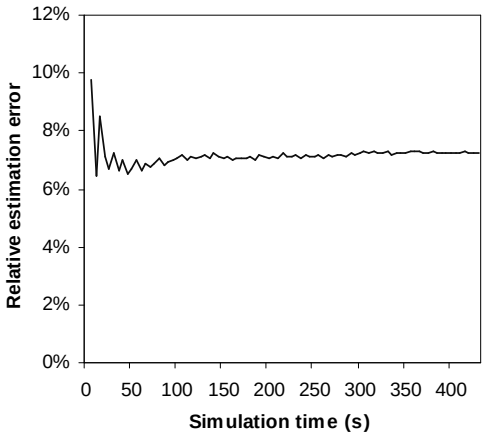
| Different Cases | Different link bandwidths (Mb/s) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $C_1$ | $A_1$ | $C_2$ | $A_2$ | $C_3$ | $A_3$ | $C_4$ | $A_4$ |
| Case-I | 2 | 0.4 | 1.5 | 0.3 | 0.8 | 0.16 | 1.5 | 0.3 |
| Case-II | 2 | 0.4 | 1.5 | 0.25 | 0.8 | 0.4 | 1.5 | 0.35 |
| Case-III | 20 | 4 | 15 | 3 | 8 | 1.6 | 15 | 3 |
| Case-IV | 20 | 4 | 15 | 2.5 | 8 | 4 | 15 | 3.5 |
| Case-V | 2 | 0.4 | 0.8 | 0.4 | 1.5 | 1.25 | 2 | 1.6 |

Table XI. Relative Estimation Error $\tilde{e}$ with TCP Cross-Traffic.

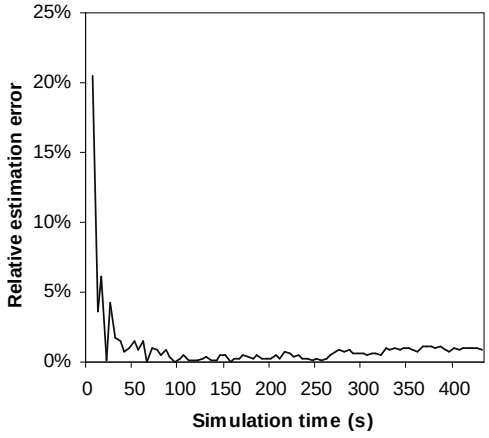
| | Relative estimation error $\tilde{e}$ | | | | |
|---|---|---|---|---|---|
| | Case-I | Case-II | Case-III | Case-IV | Case-V |
| $C_1$ | 7.25% | 8.52% | 1.69% | 3.96% | 7.76% |
| $C_2$ | 0.84% | 1.45% | 0.74% | 1.93% | 2.15% |
| $C_3$ | 2.34% | 4.05% | 1.04% | 0.02% | 39.37% |
| $C_4$ | 63.6% | 2.78% | 2.46% | 12.88% | 71.91% |
| $A$ | 5.88% | 5.45% | 3.75% | 8.28% | 7.41% |

and kept the utilization of each router R1 - R4 in the range 80-90% (the exact value depended on TCP's ability to utilize the links). The TCP cross-traffic consisted of a mixture of flows with packet sizes 600, 800, 1000 and 1200 bytes. The probe-traffic sender PT, sent packet-trains of length n = 48 with 1,500-byte packet size at an average rate of 50 kb/s. These packet-trains were enveloped by small packets of 45 bytes. The inter-packet spacing in alternate packet-trains was initialized to two different values $x_a^{(0)}$ and $x_b^{(0)}$. These values were chosen such that Lemma 1 was satisfied.

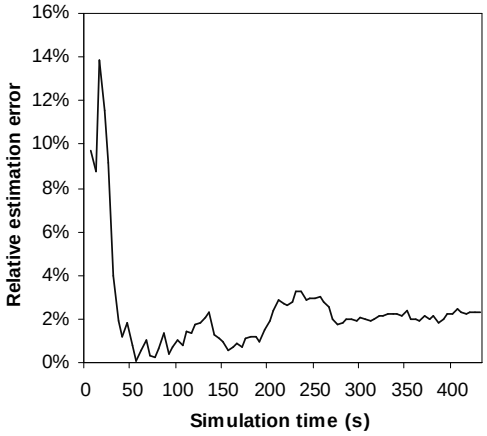
Simulation results are summarized in Table XI. As the table shows, modified Envelope correctly identifies the bottleneck router and computes C with 5% accuracy as well as available bandwidth A with 5 - 10% accuracy. In case-V where the bottleneck link precedes the tight link, capacity estimation has high errors for the links that follow the bottleneck link, which is a common problem for all measurement methods. The results are slightly better than the previous Envelope method. However, there is not much improvement in the new method over the previous in terms of the estimation accuracy. As mentioned before, this can be attributed to the fact that the error due to dispersion of leading envelope-packet gets canceled in the estimation analysis, which involves relative difference between the estimated spacing for two different signals. Hence, the simpler estimation equations and a more stable method are the enhancements over the previous method. The convergence of modified Envelope is demonstrated by the graphs in figures 31, 32, 33, 34, and 35, which also validate the asymptotic nature of the estimation method.



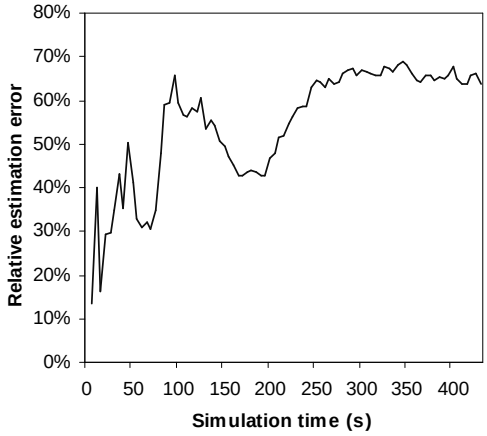

(a) $\tilde{e}_C$, $C_1 = 2$ Mb/s



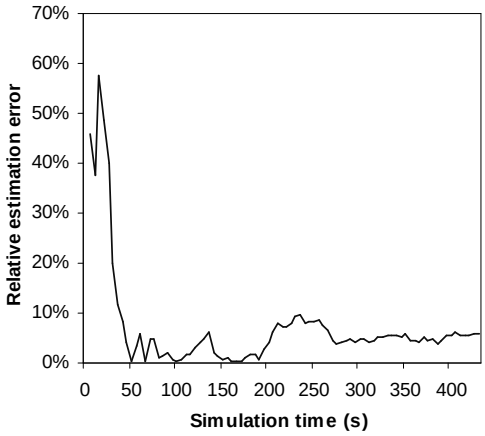

(b) $\tilde{e}_C$, $C_2 = 1.5$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 0.8$ Mb/s

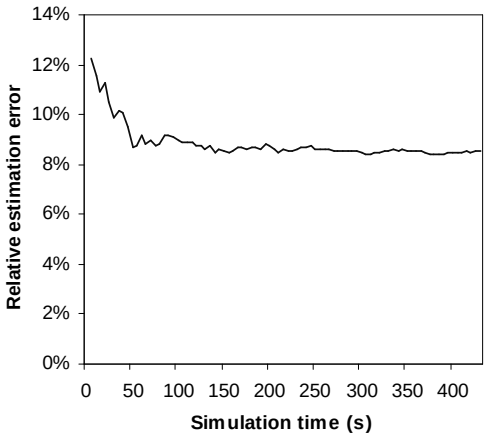



(d) $\tilde{e}_C$, $C_4 = 1.5$ Mb/s

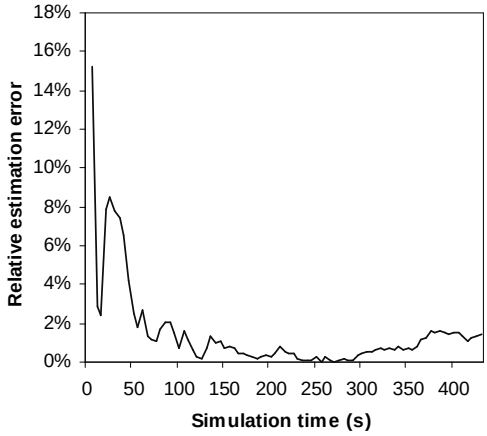



(e) $\tilde{e}_A$, $A = 160$ Kb/s in $C_3$ (c)

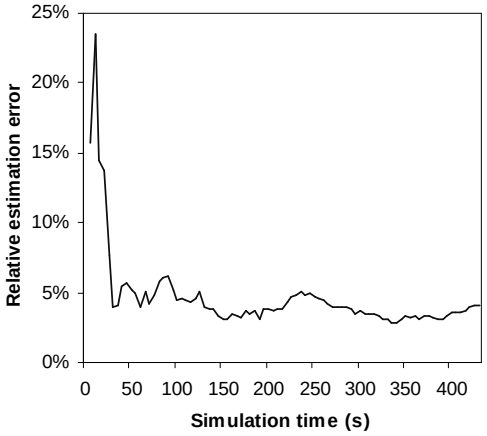
Fig. 31. TCP cross-traffic. Case-I: Bottleneck link is the tight link.



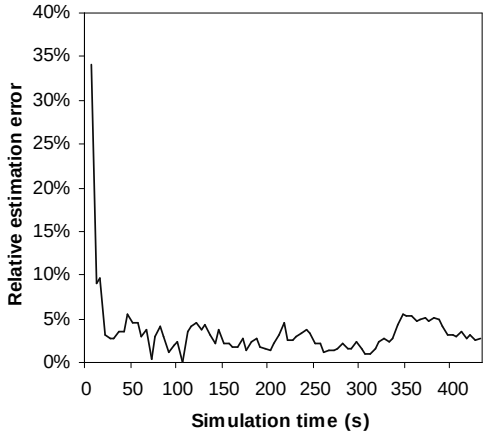

(a) $\tilde{e}_C$, $C_1 = 2$ Mb/s



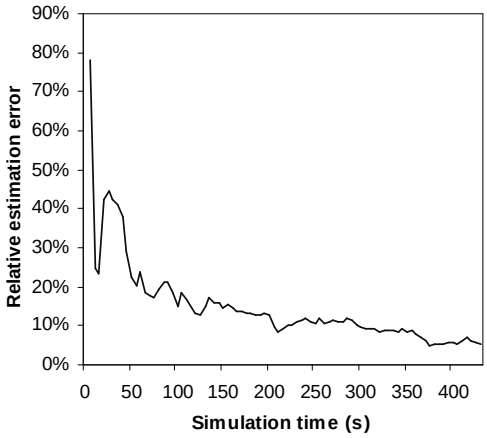

(b) $\tilde{e}_C$, $C_2 = 1.5$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 0.8$ Mb/s

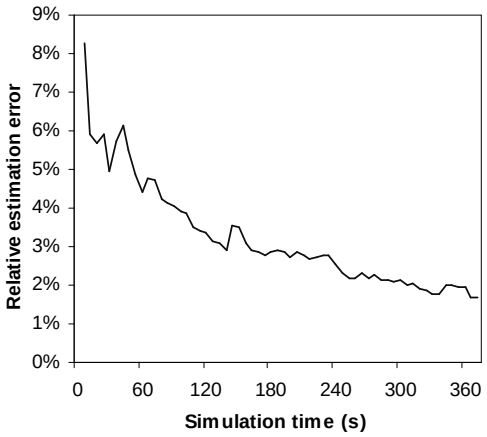



(d) $\tilde{e}_C$, $C_4 = 1.5$ Mb/s

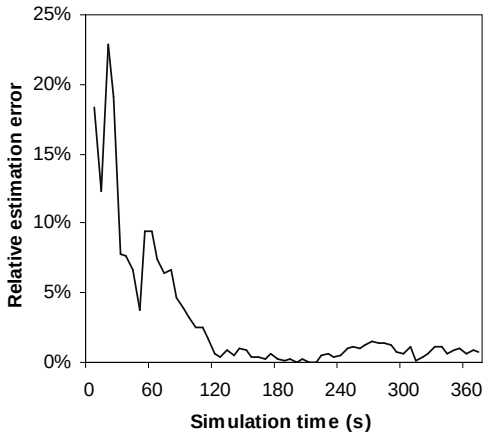



(e) $\tilde{e}_A$, $A = 250$ Kb/s in $C_2$ (f)

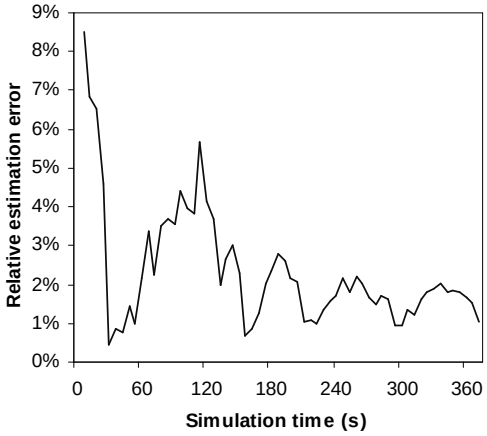
Fig. 32. TCP cross-traffic. Case-II: Bottleneck link is after the tight link.



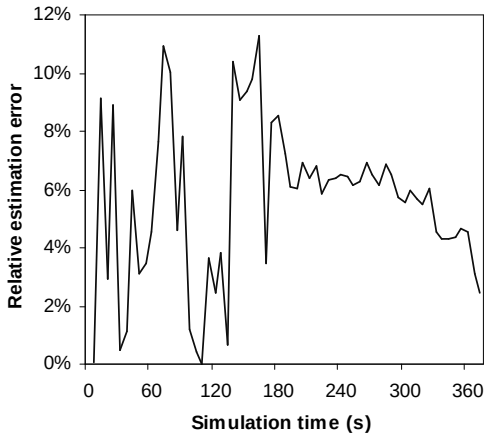

(a) $\tilde{e}_C$, $C_1 = 20$ Mb/s



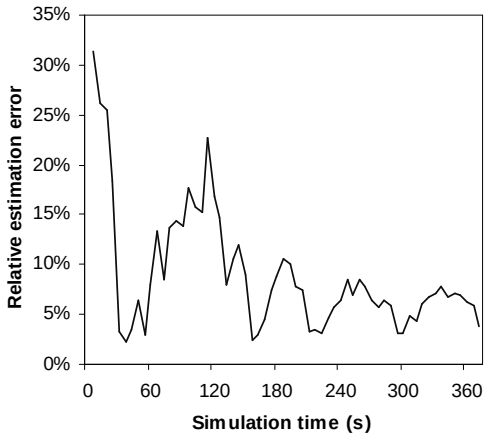

(b) $\tilde{e}_C$, $C_2 = 15$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 8$ Mb/s

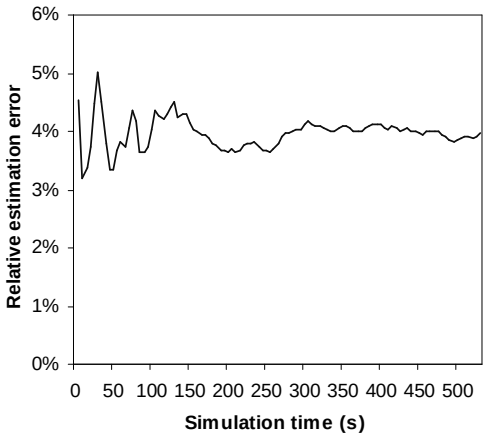



(d) $\tilde{e}_C$, $C_4 = 15$ Mb/s

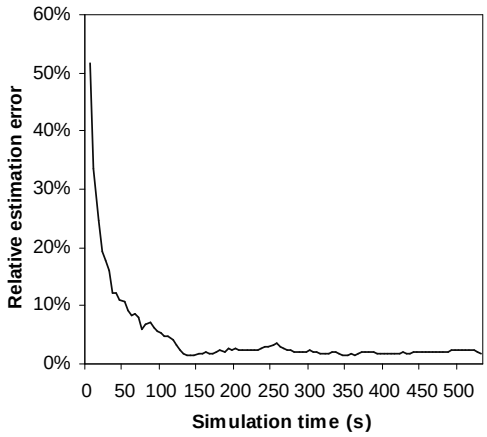



(e) $\tilde{e}_A$, $A = 1.6$ Mb/s in $C_3$ (k)

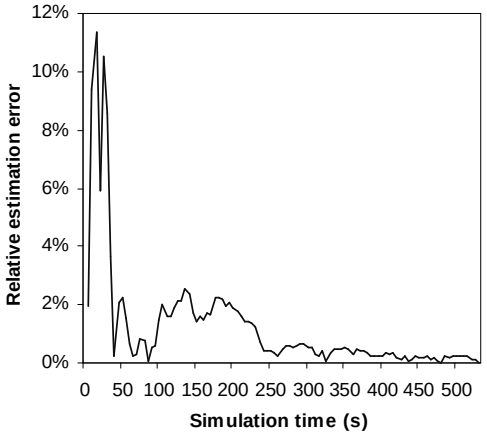
Fig. 33. TCP cross-traffic. Case-III: Bottleneck link is the tight link.



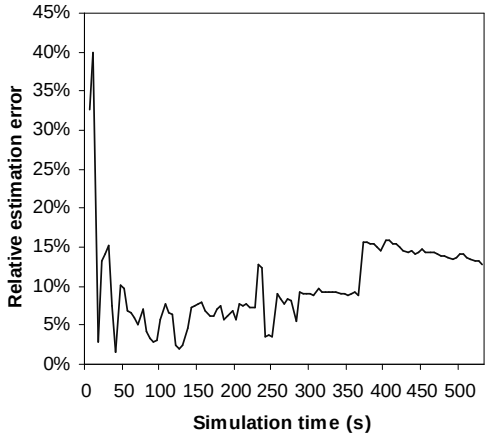

(a) $\tilde{e}_C$, $C_1 = 20$ Mb/s



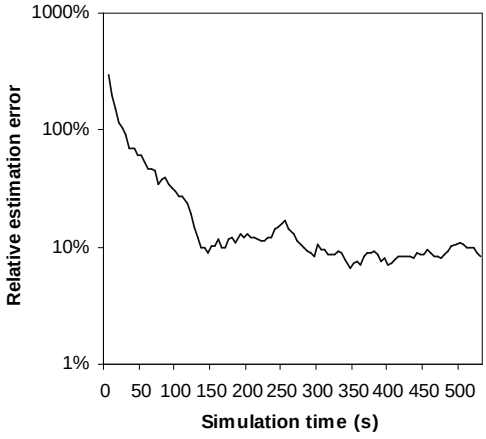

(b) $\tilde{e}_C$, $C_2 = 15$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 8$ Mb/s

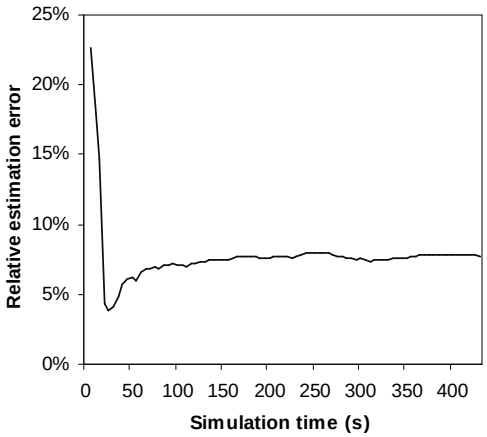



(d) $\tilde{e}_C$, $C_4 = 15$ Mb/s

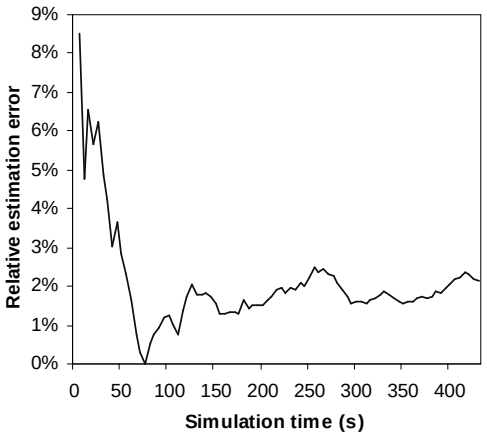



(e) $\tilde{e}_A$, $A = 2.5$ Mb/s in $C_2$ (n)

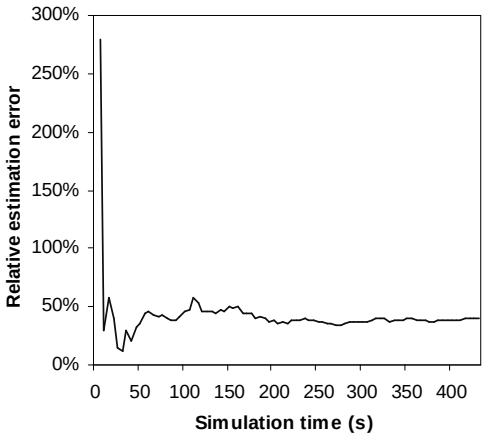
Fig. 34. TCP cross-traffic. Case-IV: Bottleneck link is after the tight link.



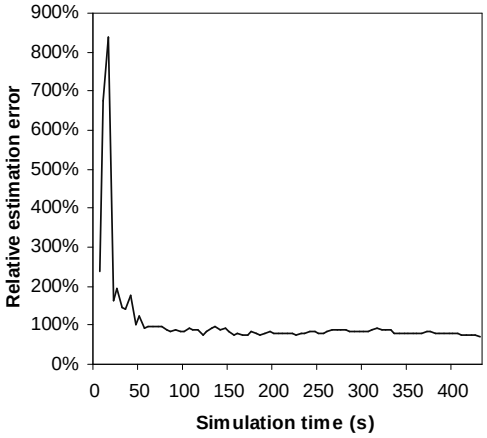

(a) $\tilde{e}_C$, $C_1 = 2$ Mb/s



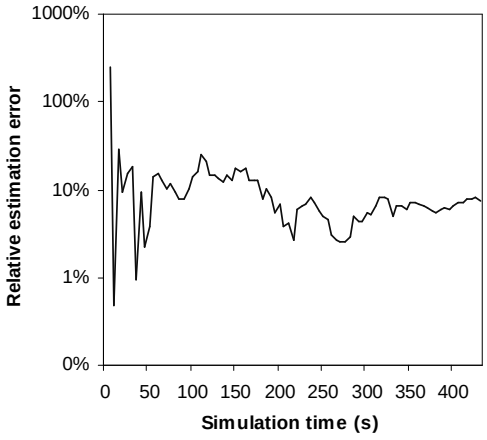

(b) $\tilde{e}_C$, $C_2 = 0.8$ Mb/s




(c) $\tilde{e}_C$, $C_3 = 1.5$ Mb/s




(d) $\tilde{e}_C$, $C_4 = 2$ Mb/s




(e) $\tilde{e}_A$, $A = 250$ Kb/s in $C_3$ (c)

Fig. 35. TCP cross-traffic. Case-V: Bottleneck link is before the tight link.

# CHAPTER VIII

## CONCLUSION

This thesis presented a new bottleneck and available bandwidth estimation technique, called Envelope, for heavily congested end-to-end paths with multiple congested nodes. Envelope is based on a queuing model (4.4) of a congested Internet router, derived by the recursive extension of the model for single congested node proposed in [9]. The simulation results show that Envelope can simultaneously estimate bottleneck and available bandwidth with good asymptotic-accuracy. In the process, it also locates the position of the bottleneck and the tight link. We also compared Envelope with methods such as Pathload, IGI, Spruce, and CapProbe using the relative estimation error $\tilde{e}$ metric. The comparison results suggests that Envelope outperforms all the methods under heavily congested paths in terms of $\tilde{e}$. In some cases, Pathload results were comparable to that of Envelope.

Simulations showed that Envelope can estimate capacity for all the hops in the path. However, at times we find that once the bottleneck router is reached, the capacity estimation of successive routers become inaccurate. This limits Envelope in locating the bottleneck link if there are two such links with almost the same capacity. The same applies to the tight links also.

The future work is focused on building a tool based on the recursive model and the estimation technique Envelope, which can do measurements over the different end-to-end Internet paths. With the tool, we would also like to compare convergence speed and path intrusiveness of Envelope with other existing methods in different network conditions.

# REFERENCES

[1] V. S. Pai, P. Druschel, and W. Zwaenepoel, "Flash: An Efficient and Portable Web Server," in *Proc. USENIX 1999 Annual Technical Conference*, pp. 199-212, Monterey, CA, June 6–11, 1999.

[2] Red Hat Inc. *TUX 2.2 Reference Manual*, 2002. Available at http://www.redhat.com/docs/manuals/tux/TUX-2.2-Manual.

[3] A. Shukla, L. Li, A. Subramanian, P. A. S. Ward, and T. Brecht, "Evaluating the Performance of User-Space and Kernel-Space Web Servers," in *Proc. IBM CASCON*, Markham, Canada, October 2004.

[4] M. Welsh, D. Culler, and E. Brewer. "SEDA: An Architecture for Well-conditioned, Scalable Internet Services," in *Proc. Eighteenth ACM Symposium on Operating Systems Principles*, pp. 230-243, Banff, Alberta, Canada, October 2001.

[5] R. S. Prasad, M. Murray, C. Dovrolis, and K. Claffy, "Bandwidth Estimation: Metrics, Measurement Techniques, and Tools," *IEEE Network Magazine*, vol. 17, issue 6, pp. 27-35, November-December 2003.

[6] M. Jain and C. Dovrolis, "End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput," in *Proc. ACM SIGCOMM*, vol. 32, issue 4, pp. 295-308, Pittsburgh, PA, August 2002.

[7] K. Lai and M. Baker, "Measuring Bandwidth," in *Proc. IEEE INFOCOM*, vol. 1, pp. 235-245, New York, NY, March 1999.

[8] V. Jacobson, "Congestion Avoidance and Control," in *Proc. ACM SIGCOMM*, vol. 18, issue 4, pp. 314-329, Stanford, CA, August 1988.

[9] S. Kang, X. Liu, M. Dai and D. Loguinov, "Packet Pair Bandwidth Estimation: Stochastic Analysis of a Single Congested Node," in *Proc. IEEE ICNP*, pp. 316-325, Berlin, Germany, October 2004.

[10] K. Harfoush, A. Bestavros, and J. Byers, "Measuring Bottleneck Bandwidth of Targeted Path Segments," in *Proc. IEEE INFOCOM*, vol. 3, pp. 2079-2089, San Francisco, CA, March 2003.

[11] N. Hu, L. Li, Z. M. Mao, P. Steenkiste, and J. Wang, "Locating Internet Bottlenecks: Algorithms, Measurements, and Implications," in *Proc. ACM SIGCOMM*, vol. 34, issue 4, pp. 41-54, Portland, OR, August 2004

[12] ns-2: The Network Simulator, http://www.isi.edu/nsnam/ns/.

[13] N. Hu and P. Steenkiste, "Evaluation and Characterization of Available Bandwidth Probing Techniques," in *Proc. IEEE JSAC: Special Issue in Internet and WWW Measurement, Mapping, and Modeling*, vol. 21, issue 6, pp. 879-974, August 2003.

[14] J. Strauss, D. Katabi and F. Kaashoek, "A Measurement Study of Available Bandwidth Estimation Tools," in *Proc. ACM SIGCOMM Internet Measurement Conference*, pp. 39-44, Miami, FL, October 2003.

[15] M. Jain and C. Dovrolis, "Pathload: A Measurement Tool for End-to-End Available Bandwidth," in *Proc. Passive and Active Measurements (PAM) Workshop*, pp. 14-25, Fort Collins, CO, March 2002.

[16] R. Kapoor, L. Chen, A. Nandan, M. Gerla and M. Y. Sanadidi, "CapProbe: A Simple and Accurate Capacity Estimation Technique for Wired and Wireless

Environments," in *Proc. ACM SIGMETRICS/Performance*, vol. 32, issue 1, pp. 390-391, New York, NY, June 2004.

[17] S. Keshav, "A Control-Theoretic Approach to Flow Control," in *Proc. ACM SIGCOMM*, vol. 21, issue 4, pp. 3-15, Zürich, Switzerland, September 1991.

[18] R. L. Carter and M. E. Crovella, "Measuring Bottleneck Link Speed in Packet Switched Networks," *International Journal on Performance Evaluation*, vol. 27/28, pp. 297-318, Lausanne, Switzerland, October 1996.

[19] V. Paxson, "Measurements and Analysis of End-to-End Internet Dynamics," Ph.D. dissertation, Computer Science Department, University of California at Berkeley, Berkeley, CA, April 1997.

[20] M. Allman and V. Paxson, "On Estimating End-to-End Network Parameters," in *Proc. ACM SIGCOMM*, vol. 29, issue 4, pp. 263-274, Cambridge, MA, August 1999.

[21] B. Ahlgren, M. Björkman, and B. Melander, *Network Probing Using Packet Trains*, Swedish Institute of Computer Science Technical Report, Kista, Sweden, March 1999.

[22] B. Melander, M. Björkman, and P. Gunningberg, "A New End-to-End Probing and Analysis Method for Estimating Bandwidth Bottlenecks," in *Proc. IEEE GLOBECOM*, vol. 1, pp. 415-420, San Francisco, CA, November 2000

[23] C. Dovrolis, P. Ramanathan, and D. Moore, "What Do Packet Dispersion Techniques Measure?" in *Proc. IEEE INFOCOM*, vol. 2, pp. 905-914, Anchorage, AK, April 2001.

[24] Y. Zhang, N. Duffield, V. Paxson and S. Shenker, "On the Constancy of Internet Path Properties," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, pp. 197-211, San Francisco, CA, November 2001.

[25] V. J. Ribeiro, R. H. Riedi, and R. G. Baraniuk, "Spatio-Temporal Available Bandwidth Estimation with STAB," in *Proc. ACM SIGMETRICS/Performance*, vol. 32, issue 1, pp. 394-395, New York, NY, June 2004.

[26] D. Zhang, W. Huang and C. Lin, "Locating the Tightest Link of a Network Path," in *Proc. ACM SIGMETRICS/Performance*, vol. 32, issue 1, pp. 402-403, New York, NY, June 2004.

[27] J. Postel, *Internet Control Message Protocol*, DARPA Network Working Group Report RFC-792, USC Information Sciences Institute, September 1981.

[28] R. W. Wolff, *Stochastic Modeling and the Theory of Queues*. Upper Saddle River, NJ: Prentice Hall, 1989

# VITA

**Name:** Amit Bhati

**Address:** 301 Harvey R. Bright Building, Computer Science Department, Texas A&M University, College Station, TX 77843-3112

**Education:**

- Master of Science, Computer Science, Texas A&M University, December 2004
- Bachelor of Technology, Computer Science and Engineering, Kakatiya University, August 2001

**Work Experience:**

- Graduate Research Assistant, Texas Transportation Institute, Texas A&M University System, September 2003 - December 2004
- Summer Intern, Genentech Inc., South San Francisco, CA, June 2003 - August 2003
- Graduate Research Assistant, Distribution Research and Education, ETID, Texas A&M University System, January 2003 - May 2003
- Application Engineer, Oracle Corporation, Hyderabad, India, September 2001 - July 2002